VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

PREDIKCE POVAHY SPAMOVÝCH KRÁTKÝCH TEXTŮ TEXTOVÝM KLASIFIKÁTOREM

MACHINE LEARNING TEXT CLASSIFIER FOR SHORT TEXTS CATEGORY PREDICTION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE **Bc. KAREL DRÁPELA**
AUTHOR

VEDOUCÍ PRÁCE **Mgr. Bc. HANA PLUHÁČKOVÁ,**
SUPERVISOR

BRNO 2018

Zadání diplomové práce/20199/2017/xdrape02

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav inteligentních systémů Akademický rok 2017/2018

Zadání diplomové práce

Řešitel: **Drápela Karel, Bc.**
Obor: Bezpečnost informačních technologií
Téma: **Predikce povahy spamových krátkých textů textovým klasifikátorem**
Machine Learning Text Classifier for Short Texts Category Prediction
Kategorie: Počítačové sítě

Pokyny:

1. Prostudujte existující algoritmy a knihovny pro strojové učení (MLlib, SKlearn).
2. Rozdělte poskytnuté datasety krátkých textových zpráv od firmy Mavenir na trénovací a testovací sadu a ručně klasifikujte trénovací sadu do sémantických kategorií vyplývajících z povahy textu dle vlastního uvážení.
3. Navrhněte aplikaci, která aplikuje vybrané algoritmy strojového učení na trénovací sadu, ze které se naučí zvolené kategorie a automaticky klasifikuje dosud nerozhodnuté krátké texty do odpovídajících kategorií.
4. Implementujte navržený algoritmus tak, aby výstupem aplikace byl CSV soubor obsahující poskytnuté krátké texty spolu s přiřazenými kategoriemi.
5. Otestujte vybraný klasifikátor na reálných krátkých textech.
6. Vyhodnoťte přesnost takto vytvořeného prediktivního modelu, navrhněte a implementujte metody zvyšující přesnost zvoleného automatického klasifikátoru.

Literatura:

- Dle pokynů vedoucího práce.

Při obhajobě semestrální části projektu je požadováno:

- Body 1 a 2.

Podrobné závazné pokyny pro vypracování diplomové práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci dřívějších projektů (30 až 40% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Pluháčková Hana, Mgr. Bc.**, UITS FIT VUT
Datum zadání: 1. listopadu 2017
Datum odevzdání: 23. května 2018

L.S.

doc. Dr. Ing. Petr Hanáček
vedoucí ústavu

Abstrakt

Práce se zabývá kategorizací krátkých spamových textů v SMS zprávách. V první části práce jsou shrnuty aktuální přístupy k textové klasifikaci a následuje popis nejpoužívanějších klasifikátorů. V dalších kapitolách je rozebrána anotace testovacích dat, implementace programu a výsledky klasifikace. Program je schopen klasifikovat texty na základě definovaných kategorií a také odhadnout přesnost klasifikátoru na trénovací sadě. Pro dva navržené typy kategorií dosahuje klasifikátor přesnosti až 82 % a 92 %. Předzpracování i výběr příznaků měly na přesnost pozitivní vliv. Přesnost je dále možné zvýšit odstraněním části vzorků, které má klasifikátor největší problémy zařadit. Při 80% pokrytí je možné zvýšit přesnost o 8-10 %.

Abstract

This thesis deals with categorization of short spam texts from SMS messages. First part summarizes current methods for text classification and it's followed by description of several commonly used classifiers. In following chapters test data analysis, program implementation and results are described. The program is able to predict text categories based on predefined set of classes and also estimate classification accuracy on training data. For the two category types, that I designed, classifier reached accuracy of 82 % and 92 % . Both preprocessing and feature selection had a positive impact on resulting accuracy. It is possible to improve this accuracy further by removing portion of samples, which are difficult to classify. With 80% recall it is possible to increase accuracy by 8-10 %.

Klíčová slova

strojové učení, textová klasifikace, spam, předzpracování, výběr příznaků

Keywords

machine learning, text classification, spam, preprocessing, feature selection

Citace

DRÁPELA, Karel. *Predikce povahy spamových krátkých textů textovým klasifikátorem.* Brno, 2018. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Mgr. Bc. Hana Pluháčková,

Predikce povahy spamových krátkých textů textovým klasifikátorem

Prohlášení

Prohlašuji, že jsem tento semestrální projekt vypracoval samostatně pod vedením paní Mgr. Bc. Hany Pluháčkové. Další informace mi poskytl Ing. Petr Šalomoun. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Karel Drápela
20. května 2018

Poděkování

Děkuji paní Mgr. Bc. Pluháčkové za odbornou pomoc a vedení této práce. Dále děkuji Ing. Petru Šalomounovi za konzultace.

Obsah

Kapitola 1

Úvod

Už od počátků internetové komunikace byl její nedílnou součástí spam. Dnes je spam přítomný na všech komunikačních platformách od e-mailu po sociální sítě. S nástupem mobilních telefonů začal rychle růst také objem spamu v SMS zprávách, které si příjemce s velkou pravděpodobností přečte a jejichž zasílání je dnes již relativně levné. U mobilních telefonů se také rozšířilo zasílání reklam přes prémiové kanály. Na těch je zaslání dražší než u klasických SMS zpráv. Někteří inzerenti proto místo prémiového kanálu, který je na tento typ zpráv určen, používají pro rozesílání reklamních kampaní kanály určené pro normální zprávy mezi uživateli mobilní sítě. V případě detekce takových zpráv je snahou operátora mobilní sítě zjistit, kdo je rozesílá, a domluvit s ním kontrakt pro zasílání zpráv na správném kanálu.

Firma Mavenir se mimo jiné zabývá detekcí kampaní v SMS zprávách. Systém na základě podobnosti obsahu porovnávané zprávy s uloženými vzorky v databázi určí, k jaké kampani zpráva patří. Jelikož je ale velmi důležité, aby normální zprávy nebyly chybně klasifikovány, systém musí být ve svých výsledcích spíše konzervativní. Může se tedy například stát, že zprávy ze stejné kampaně budou identifikovány jako různé kampaně. Cílem této práce je vytvořit systém, který umí kategorizovat krátké zprávy pomocí metod strojového učení. Na základě manuálně kategorizovaného vzorku dat by potom systém klasifikoval zbytek dat a bylo by možné dosáhnout přesnější klasifikace SMS kampaní.

Prvním cílem mé práce je popsat metody, které se používají pro textovou klasifikaci. Na základě těchto poznatků potom implementuji program pro klasifikaci krátkých spamových zpráv. Závěrečnou část práce budou tvořit experimenty, kde porovnám výkonnost jednotlivých metod a zvolím vhodné výchozí nastavení programu.

Po úvodní kapitole následuje popis problematiky spamu a zejména SMS kampaní. Ve třetí kapitole je úvod do oblasti zpracování přirozeného jazyka a dolování textových dat a následuje přehled metod pro klasifikaci textu pomocí metod strojového učení. Popsány jsou metody předzpracování, extrakce příznaků, problém nevyváženosti dat a princip metod pro selekci příznaků. Ve čtvrté kapitole jsou představeny nejpoužívanější klasifikátory pro textová data. Jedná se o Support vector machines (dále SVM), metody založené na rozhodovacích stromech a naivní Bayesův klasifikátor. V páté kapitole popisuji zpracování datové sady se SMS kampaněmi a návrh kategorií pro klasifikaci. V šesté kapitole je popsána implementace programového řešení. Následně v šesté kapitole prezentuji dosažené výsledky.

Kapitola 2

Spam

Spam je nevyžádané sdělení, které je elektronicky hromadně šířené. Obvykle hlavní motivací pro zasílání spamu je šíření reklamy nebo škodlivých programů, případně snaha získat od uživatele určité údaje (většinou osobní nebo přihlašovací do nějaké důležité služby). Opakem spamu je *ham*, což je zpráva uživatelem žádaná.

Nejrozšířenější formou spamu je stále e-mailový spam. Jeho zasílání je pro útočníky výhodné, protože se jedná o velice levnou formu reklamy, kde stačí mít k dispozici pouze seznamy e-mailů a přístup k internetu, případně zakoupený botnet (síť počítačů infikovaných virem, které jsou řízeny z jednoho centra) pro zasílání spamu ve větším objemu. Obrana proti spamu je mnohem náročnější a dražší, než jeho šíření.

Objem zasílaného spamu se od počátku internetové komunikace rychle zvyšoval. V roce 1997 tvořil e-mailový spam 10 % internetové komunikace, což se zvýšilo na až 72 % v roce 2012 [10]. Naopak v posledních letech se podle statistik antivirové firmy Kaspersky, která se mimo jiné zabývá i detekcí spamu, objem spamu snižuje. Podle jejich ročního přehledu o spamu a phishingu se objem spamu v roce 2012–2016 snížil z 72.1 % na 58.31 % v roce 2016 [10]. Firma Kaspersky to připisuje postupnému zlepšování antispamových filtrů, které jsou stále efektivnější a v dnešní době umí zastavit více než 98 % spamové komunikace. Podle Wanga [26] se šiřitelé spamu v poslední době začínají zaměřovat spíše na sofistikovanější spamové zprávy, které obejdou spamové filtry, než na rozesílání co největšího objemu zpráv.

2.1 SMS spam

Se snížením objemu e-mailového spamu se zvyšuje podíl spamu na jiných službách. Jedná se zejména o SMS zprávy a internetové služby jako blogy, chatovací klienti nebo sociální služby (Twitter, Facebook). SMS spam stále zůstává v počtu přenášených zpráv za e-mailovým spamem. Výhodou SMS spamu je vyšší cílenost. Lidé si téměř vždy čtou došlé SMS zprávy, zatímco u e-mailů to ne vždy platí. Dalším problémem je, že uživatelé ještě nejsou na SMS spam navyklí a považují mobilní zprávy za důvěryhodnější než například e-mailovou komunikaci. Navíc jsou i více ochotní si přes SMS vyměňovat citlivé údaje. Problematické je i to, že u některých operátorů je nutné za přijaté SMS zprávy platit. Potenciálních cílů SMS spamu je velké množství. Odhaduje se, že SMS zprávy využívá 3,5 miliardy uživatelů, což je přes 80 % všech uživatelů mobilních zařízení [22]. Podíl SMS spamu se také liší podle oblasti. V Severní Americe zůstává SMS spam pod jedním procentem z celkového objemu zpráv, zatímco v Asii SMS spam tvoří až 30 % komunikace. To je způsobeno velmi nízkou cenou za zaslání SMS zprávy v této oblasti. V roce 2013 stálo zaslání jedné SMS v Číně méně než

$0,001. Používané algoritmy pro detekci spamu mohou mít nižší efektivitu pro SMS zprávy, protože ty bývají obvykle mnohem kratší a obvykle se používá jiný styl vyjadřování, než spam na jiných platformách. Další příbuznou kategorií je šíření spamu přes MMS, kde už nejsou útočníci omezeni jen na text, ale mohou posílat většinu toho, co je možné posílat přes e-mailové přílohy.

Zprávy jsou filtrovány buď na základě obsahu nebo na základě statistických příznaků. Obsahový přístup se snaží identifikovat slova nebo skupiny písmen, které se ve spamech často objevují, zatímco v hamech se vyskytují jen zřídka. Nevýhodou je, že je nutný přístup k samotnému obsahu SMS zprávy, kde vyvstávají otázky o ochraně soukromí. Statistické metody používají údaje jako počty zpráv zaslaných z daného čísla, jak rychle a odkud jsou zprávy posílány, frekvence reakcí na zprávu, velikost zpráv a další [29]. Zde není nutný přístup k obsahu zprávy a detekce se provádí na základě agregovaných statistik.

Spam není jedinou možnou formou šíření reklamy přes SMS zprávy. Většina mobilních operátorů nabízí tzv. prémiové SMS zprávy, přes které firmy mohou šířit reklamu na svoje služby. Výhodou je vysoká cílenost reklamního sdělení a vysoká pravděpodobnost, že reklama bude přečtena. Dalším pozitivem je, že vysoké procento lidí rádo přijímá reklamní SMS zprávy. Podle Okazakiho [17] je to až 60 % uživatelů. Reklamní SMS mají speciální sazby, které jsou vyšší, než stojí zaslání normální SMS zprávy mezi uživateli. Určité firmy proto zkouší reklamu zasílat přes obyčejné mobilní telefony, kde jsou ceny nižší. V těchto zprávách už často není možné příjem odhlásit a je možné je klasifikovat jako spam.

Kapitola 3

Kategorizace textu a metody pro extrakci příznaků

Zpracování přirozeného jazyka a dolování v textových datech mají širokou řadu aplikací a podoborů. Spojujícím prvkem je práce s textovými daty. Stále více se uplatňuje klasifikace citového zabarvení (anglicky sentiment) pro určení, zda je sdělení pozitivní nebo negativní. Tato informace se potom může uplatnit, v procesu sledování nálad na sociálních sítích a monitorování reakcí na reklamu nebo produkt. Obor získávání informací (information retrieval) se snaží z množiny zdrojů extrahovat relevantní informace, které uživatel potřebuje. Metody z tohoto oboru potom používají vyhledávače, knihovny a další instituce, které se zabývají vyhledáváním, uchováním a zpracováním informací. Metody určování autorství se používají ve forenzní lingvistice pro detekci pachatelů. Příbuzným oborem je detekce plagiátů, kde je cílem zjistit, zda daný text byl částečně opsán nebo převzat z jiného zdroje. Další aplikací je detekce vlastností na základě textů daného člověka. Kromě povahových vlastností se často detekují i další údaje jako věk, pohlaví nebo země původu.

Kategorizace má dvě hlavní skupiny aplikací:

- Organizace dokumentů: firmy, které musejí každý den zpracovat velké množství textových dat (např. novinové redakce využívají kategorizaci k roztřídění dokumentů pro jednodušší zpracování).
- Filtrování: jsou nastavené kategorie, které chceme přijmout, a kategorie, které filtrem neprojdou. Využívá se hlavně u filtrace spamu.

V této práci se budu věnovat kategorizaci textů do skupin podle tématu. Významným problémem při textové klasifikaci je vysoká dimenzionalita, kterou se vyznačuje většina používaných modelů. To má za následek pomalé učení a klasifikaci a také vyvolá potřebu velkého množství trénovacích dat pro naučení modelu. Pro řešení tohoto problému se používá předzpracování, filtrace a metody výběru příznaků. V následující kapitole jsou tyto skupiny metod popsány.

Další problém, který je potřeba řešit, jsou vícejazyčné texty. Takové texty jsou typické pro data získaná z internetových služeb, kde mohou přispívat uživatelé z celého světa. Jednotlivé jazyky a jazykové rodiny (indoevropská, čínská, arabská) se často výrazně liší a určité metody mohou mít různou účinnost, pokud jsou aplikovány na různé jazyky. Je proto velkou výhodou, pokud je metoda nezávislá na používaném jazyce. Specifikem textů z internetu je slovník s množstvím zkratek, nových slov, emotikonů a dalších speciálních

znaků. Obsahují také mnohem vyšší podíl šumu, zejména gramatických chyb a překlepů. Texty často bývají velmi krátké (tweety o délce 140/280 znaků).

3.1 Předzpracování

Předzpracováním v oblasti dolování dat a strojového učení se rozumí taková fáze zpracování dat, která se provádí ještě před extrakcí příznaků, a provádí transformace, jež mají za úkol zjednodušit data a zbavit je šumu, ale současně se minimalizuje ztráta informace. Formát výstupu je stejný jako formát vstupu (v případě textových dat tedy text). Jelikož se předzpracování provádí jako jeden z prvních kroků při zpracování dat, má významný vliv na další fáze. V oblasti textové klasifikace je tato fáze obzvlášť důležitá, protože textové soubory jsou obvykle vyjadřovány ve vysoce dimenzionálních prostorech a redukce této dimenzionality je důležitá zejména pro rychlost konvergence metod strojového učení.

Pro předzpracování textových dat existuje řada metod. Jejich efektivita závisí zejména na klasifikační úloze a datech, se kterými se pracuje. Dalším důležitým faktorem je také pořadí, v jakém jsou jednotlivé kroky provedeny. Určité permutace výsledek nijak neovlivní (jestli se nejdříve odstraní čísla nebo interpunkční znaménka), jiné ale mohou mít vliv na výsledek.

3.1.1 Stopslova

Stopslova označují slova, která se v textu vyskytují často, ale nenesou žádnou významovou informaci a mají zpravidla pouze syntaktický význam. Stopslov bývá vzhledem k celkovému počtu slov v jazyce poměrně málo, ale vyskytují se v textu velmi často, protože jsou obvykle nezbytná pro sestavení věty. Zejména se jedná o předložky, spojky a zájmena. V obecném smyslu ale stopslova nejsou jasně vymezena a záleží hlavně na oblasti aplikace – například pokud pracujeme s korpusem článků o počítačích, tak slovo „počítač“ je možné přidat do seznamu stopslov.

Existují dva hlavní přístupy pro získání seznamu stopslov. První je, že se seznam vypočítá přímo z datové sady, se kterou pracujeme. Například Ferilli [8] navrhuje stopslova identifikovat jako slova, která se v dokumentech v korpusu vyskytují výrazně častěji než průměrná slova. Spočítá se práh podle vzorce 3.1, kde n je počet termů v dané datové sadě, t_i je frekvence i-tého termu a α je faktor přizpůsobivosti, kterým lze práh zvýšit nebo snížit. Jako stopslova se berou slova s hodnotou vyšší než zadaný práh.

$$\sigma = \frac{\alpha}{n} \sum_{i=1}^{n} t_i \tag{3.1}$$

Druhým přístupem je využití předem zkompilovaného slovníku nejčastějších stopslov v daném jazyce. Tento typ seznamů stopslov už obsahuje pouze stopslova nezávislá na oblasti jejich aplikace, především již zmíněné předložky, spojky a zájmena.

3.1.2 Čísla, odkazy

Číselné a časové údaje nejsou z hlediska klasifikace textu samy o sobě příliš informativní. Často jsou pro danou zprávu unikátní a tím je jejich informační přínos z hlediska klasifikace malý. Existují i výjimky, kdy číselné údaje jsou důležité, například odkazy na jednotlivé

paragrafy v právnických dokumentech nám mohou napovědět hodně o obsahu textu [6]. Obvykle je ale zvykem nahradit číselné a časové údaje předem daným tokenem, který pouze vyjadřuje informaci, že zde bylo nějaké číslo. Informace o počtu číselných údajů může být užitečná například pro rozlišení faktur mezi dokumenty datové sady, ale velikosti jednotlivých částek už podstatné nejsou.

URL odkazy už v sobě obsahují určitou informaci o tématu textu, každopádně pro jejich vysokou variabilitu jsou obvykle ve fázi předzpracování odstraňovány nebo nahrazovány tokenem značícím URL odkaz. U kratších textů, kde většinu obsahu tvoří právě odkaz, je možné ale přijít o důležitou informaci.

3.1.3 Stematizace a lemmatizace

Stematizace znamená nalezení kmene (anglicky stem) slova. Kmen je ta část slova, která se u různých tvarů tohoto slova nemění. Tato operace nám opět umožňuje zjednodušit tokenizaci a zredukovat slovník, se kterým budou muset dále pracovat metody výběru příznaků a strojového učení. V praxi se ukazuje, že z hlediska významu slova je nejdůležitější jeho kořen. Proto je možné v textu ponechat pouze kořeny slov a informace o významu slov zůstane z větší části zachována.

Hlavní problémy při stematizaci se dělí na overstemming a understemming. Overstemming znamená, že dvě slova s různým kmenem se stematizují na stejný kmen. Například ve slovních spojeních *college students partying* a *political parties* se *partying* i *parties* stematizují na *parti*. Understemming nastane, když dvě slova, která sdílí společný kmen, nejsou na jeden kmen stematizována. V praxi se odlišují lehké a těžké stematizátory. Lehké snižují overstemming, ale zvyšují understemming, těžké zase naopak snižují understemming a zvyšují overstemming [11].

Přínos stematizace je hlavně ve snížení počtu unikátních slov, která se v textu vyskytují [6]. To někdy může i zvýšit přesnost klasifikace. Stematizace se používá i pro zlepšení pokrytí klasifikace [11] (počtu správně klasifikovaných vzorků dané třídy vzhledem ke všem vzorkům, o kterých si klasifikátor myslel, že patří do této třídy). Podstatné je, na jaká data je stematizace aplikována. To potvrzuje i práce [23], kde se zjistilo, že na datové sadě *Reuters*[1] byla stematizace spíše na škodu, zatímco na na datové sadě *20 Newsgroups*[2] pomohla dosáhnout přesnějšího výsledku. Na druhou stranu Pomikálek [19] nezaznamenal na stejných datech žádné významné zlepšení se stematizovanými texty. Také Darwish [5] nezjistil žádné statisticky významné zlepšení po aplikaci stematizace na arabských textech z mikroblogů.

Podobným procesem jako stematizace je lemmatizace. Ta nehledá kmen slova, ale snaží se slovo převést na jeho základní tvar. Na rozdíl od většiny stematizátorů bere v úvahu i kontext slova a slovní druhy. Lemmatizace je výrazně náročnější než stematizace, jak časově, tak algoritmicky. Výsledky jsou ale přesnější, protože například slova *introduction*, *introducing*, *introduces* stematizér ořízne na *introduc*, zatímco lemmatizér je převede na základní tvar – *introduce*. Podobně u stematizace se slova *go* a *went* nezkrátí na stejný kořen, zatímco u lemmatizace budou obě převedena na základní tvar *go*. Lemmatizace se používá u textové klasifikace méně než stematizace, hlavně z důvodu její vyšší časové i výpočetní složitosti, která se většinou neodrazí ve výrazně lepších výsledcích klasifikace.

[1] https://archive.ics.uci.edu/ml/machine-learning-databases/reuters21578-mld/
[2] http://qwone.com/~jason/20Newsgroups/

3.1.4 Slovní druhy

K jednotlivým slovům textu je možné určit slovní druhy, které klasifikačnímu algoritmu poskytnou další informaci. Textově stejná slova mohou mít různé slovní druhy v závislosti na kontextu, ve kterém se nachází. Slovní druhy umožňují zachovat informaci, která se mohla ztratit některými metodami předzpracování – například stematizací. Na základě slovních druhů je také možné váhovat (přiřadit stupeň důležitosti) příznaky textu. Pro určité klasifikační úlohy mají totiž určité slovní druhy vyšší důležitost. Například pro klasifikaci podle tématu jsou nejdůležitější podstatná jména a slovesa, zatímco systémy pro určování citového zabarvení se soustředí hlavně na přídavná jména a příslovce [8].

3.2 Extrakce příznaků

Extrakce příznaků je proces transformace vstupních dat na vektor příznaků, který je následně možné zpracovat klasifikačními algoritmy. Cílem je, aby výstupní vektor co nejvíce vystihoval vstupní data z hlediska řešené klasifikační úlohy a zachoval relevantní informace v datech obsažené. Textové extrakční algoritmy řeší otázku, jakým způsobem co nejefektivněji reprezentovat text. Obecně bývá reprezentován jako vektor vážených termů, kde term je základní textová jednotka, se kterou metody pracují. Může se jednat o slovo, písmeno, případně jinak vymezené části textu.

Filtrace

Úkolem filtrace je rychle odstranit ty příznaky, u kterých je vysoká pravděpodobnost, že budou mít malou diskriminativní sílu. Filtrační metoda musí být především rychlá, protože obvykle pracuje s velkým množstvím dat. Metoda neřeší závislosti mezi třídami či příznaky a proto často ponechává redundantní příznaky. To ale nevadí, protože se používá pouze jako předstupeň pro metody výběru příznaků (kapitola 3.4), které jsou přesnější, ale jejich výpočet trvá déle a není možné je aplikovat na všechny extrahované příznaky.

Základní filtrační metrikou je filtrace na základě počtu. Příznak, který se vyskytuje pouze v jednom nebo několika málo dokumentech, je možné odstranit. Stejně tak příznak, který se v přibližně stejných počtech vyskytuje ve valné většině dokumentů, nebude při klasifikaci moc užitečný.

Normalizace

Základním způsobem reprezentace příznaků je jejich počet. Příznakový vektor pro každý dokument obsahuje informaci, kolikrát se daný příznak vyskytl v dokumentu. Tato metoda funguje dobře, pokud jsou dokumenty v datové sadě přibližně stejné délky. Pokud jsou v délce dokumentů významné rozdíly, dochází ke zkreslení a algoritmy strojového učení mají tendenci dávat delším dokumentům vyšší váhu, než by měly. Pro řešení tohoto problému je často doporučována normalizace do intervalu $< 0, 1 >$. Často používanou metodou pro normalizaci příznaků je vzorec 3.2, kde t_i je hodnota příznaku i-tého prvku slovníku a w_i je příslušná váha [24].

$$fv = \frac{(w_1 \cdot t_1, w_2 \cdot t_2, ..., w_n \cdot t_n)}{\sqrt{\sum_{i=1}^{n}(w_i \cdot t_i)^2}} \tag{3.2}$$

Váhování

Jednotlivým příznakům je možné přiřadit různou váhu v závislosti na jejich relativní důležitosti. Výslednou hodnotou příznaku potom bude součin váhy a jeho původní hodnoty. Obvyklým způsobem určení váhy příznaků bývá převrácená četnost termu v dokumentu (inverse document frequency). Váha i-tého příznaku IDF_i je definována podle vzorce 3.3, kde N je počet dokumentů v datové sadě a df_i je četnost i-tého termu, tedy v kolika dokumentech se tento term vyskytuje.

$$IDF_i = \log \frac{N}{df_i} \tag{3.3}$$

Binární příznaky

Alternativním modelem jsou binární příznaky. Místo zaznamenávání počtu výskytů zaznamenáváme pouze, zda se daný příznak v dokumentu vyskytl či nikoliv. Je jasné, že zde nastává určitá ztráta informace, která je tím větší, čím jsou větší rozdíly mezi počty výskytů příznaků. Tento model se používá zejména pro reprezentaci krátkých textů, kde vektor příznaků je velmi řídký (vysoký počet nulových hodnot) a nenulové příznaky nedosahují vysokých hodnot. V těchto textech je mnohem důležitější informace, zda se daný příznak u textu vyskytl či nikoliv, než přesný počet výskytů (mezi počty výskytů jsou obvykle malé rozdíly).

3.2.1 Bag of Words model

Jedná se o jednoduchou, ale často používanou metodu pro extrakci příznaků. Bag of Words model (dále BOW) pracuje ve dvou fázích, kde v první se projde textový korpus a vytvoří se slovník všech slov, která se v textu vyskytují. Následně se korpus projde znovu a pro jednotlivé texty se vytvoří statistika počtu výskytů slov slovníku v daném textu. Je jasné, že tato metoda vytvoří velmi dlouhé vektory příznaků, jejichž délka odpovídá počtu unikátních slov. Obvykle se pro filtraci a výběr příznaků používají další metody, které odstraní méně informativní prvky a redukují tak výsledný vektor. Další nevýhodou je nutnost mít v paměti uložený slovník termů, který může být značně rozsáhlý. Pro snížení paměťových nároků je možné použít například hashování [28]. Výhodné je, že metoda není závislá na použitém jazyce. Obvykle se používá spíše pro delší texty.

3.2.2 N-gramy

N-gramy jsou zobecněním BOW modelu pro skupiny více tokenů. N-gram je posloupnost n prvků z dané posloupnosti. V oblasti extrakce příznaků se potom pracuje s posloupnostmi n tokenů, které jsou pokládány za jeden příznak textu. Prvky BOW modelu jsou potom unigramy, neboli posloupnosti délky 1. Pro vyšší hodnoty n bývají výstupem ještě objemnější a řidší vektory příznaků než je tomu u BOW modelu a opět je nutné provést filtraci a výběr příznaků. N-gramy jsou jazykově nezávislé. Výhodou n-gramů (pro $n > 1$) je, že neberou v úvahu pouze jeden token, ale i jeho okolní kontext. To je důležité, protože text je ze své podstaty kontextuálně založený a jedno slovo může mít několik různých významů v závislosti na tom, v jakém kontextu je použito. Ze samotného slova je potom často nemožné správnou informaci získat a tato ztráta informace může mít kritické důsledky v určitých oblastech klasifikace textu, zejména u klasifikace podle citového zabarvení („good“, „not good“). V oblasti detekce tématu už kontext není natolik důležitý a často se používají i

unigramy. Při vysokých hodnotách n ($n > 5$) se ale efektivita n-gramů snižuje, protože model není dostatečně obecný a velikost vektoru příznaků velmi rychle roste. Pro slovní n-gramy se obvykle volí $n < 4$.

Písmenné n-gramy

Speciální kategorií jsou potom písmenné n-gramy, které jako tokeny používají jednotlivá písmena. Tato varianta je vhodná především pro kratší texty, kde slovní n-gramy nemusí být dostatečně diskriminativní, jelikož každý text obsahuje maximálně několik desítek slov. Typicky se používají pro klasifikaci zpráv na mikroblogovacích službách jako je Twitter. Oproti slovním n-gramům také umějí lépe zachytit interpunkci, překlepy a různé zkratky i emotikony typické pro internetovou komunikaci. Prakticky používané délky slovních n-gramů jsou v rozmezí od dvou do pěti písmen.

Skip-gramy

Další variantou jsou skip-gramy. Ty berou v úvahu všechny kombinace tokenů bez ohledu na jejich posloupnost. V případě slovních n-gramů bychom mohli například vytvářet skip-gramy ze slov věty, případně z celého textu. Používají se v případech, kdy samotná posloupnost tokenů není důležitá pro klasifikační úlohu. Za ztrátu informace o návaznosti tokenů se získá obecnější model reprezentace textu.

3.2.3 Další modely

Existují i další, méně používané modely pro reprezentaci textu. Grafové modely se zaměřují na co nejpřesnější zachycení kontextu, ve kterém se slovo nachází. Tenzorová reprezentace textu je zobecněním vektorové, případně maticové, na vícedimezionální reprezentaci. Potom trigramy je možné reprezentovat například v třídimenzionálním prostoru, kde každá dimenze značí jeden znak trigramu [15].

3.3 Vyvážení dat

Problém nevyváženosti dat je důležité téma v oblasti strojového učení, kterému se věnuje řada studií [7] [20] [9] [12] [25]. V ideální datové sadě by všechny třídy měly mít stejný počet vzorků, protože většina standardních klasifikátorů (neuronové sítě, rozhodovací stromy) předpokládá rovnoměrné rozložení tříd [7]. Toho je v praxi obtížné dosáhnout. Byly proto vyvinuty metody, které se snaží datovou sadu transformovat takovým způsobem, aby třídy byly vyvážené.

Třída, která se v datové sadě vyskytuje velmi často, se nazývá majoritní, zatímco ta, která se vyskytuje velmi zřídka, se nazývá minoritní. Často je problematické, že cena za nesprávnou klasifikaci minoritní třídy je mnohem vyšší než cena za nesprávnou klasifikaci majoritní třídy (například detekce rakoviny nebo detekce podvodných zpráv). Často jsou bohužel tyto ceny pro jednotlivé třídy neznámé.

V určitých případech je možné vyřešit nevyváženost dat nashromážděním více vzorků, kde se buď třídy vyváží nebo se rozdíly sníží na přijatelnou úroveň. Ne vždy je možné získat více vzorků nebo se minoritní třída opravdu reálně vyskytuje méně často než majoritní třída.

3.3.1 Vzorkovací metody

Vzorkování se provádí ve fázi předzpracování a cílem je z nevyvážené datové sady vytvořit vyváženou. To se dělá buď přidáním vzorků do minoritní třídy (nadvzorkování, oversampling) nebo odebráním vzorků z majoritní třídy (podvzorkování, undersampling). Základními algoritmy jsou náhodné podvzorkování a náhodné nadvzorkování. Zde se vezme náhodný vzorek z datové sady, který se buď zduplikuje nebo odstraní. Vzorkování se dělí na metody „s návratem" a „bez návratu". Vzorkování s návratem vždy vybírá z původní datové sady, takže jeden vzorek je možné vybrat vícekrát. Naopak u vzorkování bez návratu je každý vzorek možné vybrat maximálně jednou.

Problematické u nadvzorkování je, že zvyšuje objem dat, se kterými musí klasifikátor pracovat, čímž se může doba trénování, případně i klasifikace, výrazně zvýšit. Dalším problémem je, že u náhodného vzorkování hrozí přeučení na trénovacích datech. Naopak u podvzorkování je problémem, že odstraněním části vzorků je možné přijít o důležité vzory v datech, které by mohly potencionálně pomoci k lepším výsledkům klasifikace [7].

K řešení těchto problémů byla navržena metoda SMOTE (synthetic minority over sampling technique), která generuje syntetické vzorky pomocí pravidla nejbližšího souseda. Další metoda pro generování syntetických vzorků je ADASYN (adaptive synthetic sampling), která se zaměřuje na generování vzorků v blízkosti vzorků, které byly nesprávně klasifikovány použitím k-NN klasifikátoru. Tyto dvě metody se liší pouze způsobem výběru sousedů. Po výběru vzorku se vybere náhodně jeden soused a nový vzorek se vygeneruje v polovině spojnice původního vzorku a vybraného souseda.

3.3.2 Další metody

Učení podle ceny (cost sensitive learning) – bere v úvahu cenu za nesprávnou klasifikaci vzorku a počítá s tím, že tato cena je odlišná pro různé třídy. Přiřazuje minoritní třídě vyšší cenu a snaží se vygenerovat model s co nejnižší cenou [25].

Jednotřídní klasifikace – klasifikátor je učen pouze na rozpoznávání jedné třídy (minoritní). Tato metoda zlepšuje přesnost rozpoznání minoritní třídy, proto je vhodné ji použít, pokud je nejdůležitější rozpoznat právě minoritní třídu [20].

Souborné metody – pro zlepšení zobecňovací schopnosti využívají více klasifikátorů. Základním přístupem je boosting, kde je několik klasifikátorů v kaskádě a každý pracuje s výstupem toho předchozího [9]. Druhým je bagging, kde se použije několik klasifikátorů pracujících se stejnými daty a následně se nějakou metodou (například hlasováním) určí výsledná třída [12].

3.4 Selekce příznaků

V následující kapitole jsou popsány vybrané metody výběru příznaků. Jedná se o metody Gini index, Informační přínos, χ^2 test a metoda společné informace.

3.4.1 Gini index

Nechť $p_i(w)$ je podmíněná pravděpodobnost, že dokument náleží do třídy i vzhledem k tomu, že obsahuje slovo w a tedy platí rovnice 3.4. Potom Gini index pro slovo w, značený $G(w)$, je definován podle rovnice 3.5 [1]. Hodnota Gini indexu nabývá hodnot v intervalu $(\frac{1}{k}, 1)$. Čím je jeho hodnota vyšší, tím je vyšší rozlišovací schopnost slova w. Pokud by všechny dokumenty obsahující slovo w patřily do jedné třídy, pak by hodnota $G(w)$ byla 1.

Naopak pokud by dokumenty se slovem w byly rovnoměrně rozděleny mezi jednotlivé třídy, pak by hodnota $G(w)$ byla $\frac{1}{k}$.

$$G(w) = \sum_{i=1}^{k} p_i(w) = 1 \tag{3.4}$$

$$G(w) = \sum_{i=1}^{k} p_i(w)^2 \tag{3.5}$$

Gini index předpokládá, že v datové sadě jsou třídy rovnoměrně rozloženy. Problematické jsou tedy datové sady s nevyváženými třídami, kde výsledné indexy nemusí odpovídat spočítané diskriminativní síle slov. K řešení tohoto problému se používá normalizovaný Gini index, který přesněji vyjadřuje rozlišovací schopnost slov v datové sadě s obecnými třídami. Nechť $P_1 ... P_k$ je souhrnné rozložení dokumentů do tříd. Potom se spočítá normalizovaná pravděpodobnost $p_i^{'}(w)$ podle rovnice 3.6 [1]. Následně normalizovaný Gini index se spočítá podle rovnice 3.7. Normalizace pomocí globálního rozložení dokumentů do tříd produkuje přesnější výsledky při nevyvážených datech. Složitost metody pro n dokumentů, d slov a k tříd je $O(n \cdot k \ \cdot d)$.

$$p_i^{'}(w) = \frac{p_i(w)/P_i}{\sum_{j=1}^{k} p_j(w)/P_j} \tag{3.6}$$

$$G(w) = \sum_{i=1}^{k} p_i^{'}(w)^2 \tag{3.7}$$

3.4.2 Informační přínos

Nechť P_i je globální pravděpodobnost třídy i a $p_i(w)$ je pravděpodobnost, že dokument patří do třídy i, pokud obsahuje slovo w. $F(w)$ je podíl dokumentů obsahujících slovo w. Potom informační přínos IG pro slovo w se spočítá podle rovnice 3.8 [1]. Znovu platí, že čím vyšší je hodnota IG, tím vyšší rozlišovací schopnost má dané slovo. Časová složitost metody pro n dokumentů, d slov a k tříd je $O(n \cdot k \ \cdot d)$

$$\begin{aligned} IG(w) = \sum_{i=1}^{k} P_i \cdot \log(P_i) + F(w) \cdot \sum_{i=1}^{k} p_i(w) \cdot \log(p_i(w)) + \\ +(1 - F(w)) \cdot \sum_{k}^{i=1} (1 - p_i(w)) \cdot \log(1 - p_i(w)) \end{aligned} \tag{3.8}$$

3.4.3 Vzájemná informace

Metoda vzájemné informace (mutual information) je odvozena z teorie informace a modeluje vzájemnou informaci mezi příznaky a třídami. Očekávaný současný výskyt třídy i a slova w za předpokladu vzájemné nezávislosti je dán jako $P_i \cdot F(w)$. Skutečný současný výskyt se spočítá jako $F(w) \cdot p_i(w)$. Pokud je skutečný výskyt menší než očekávaný, je mezi nimi negativní korelace, pokud je skutečný výskyt vyšší než očekávaný, tak se jedná o pozitivní korelaci. Vzájemná informace $M_i(w)$ mezi třídou i a slovem w se spočítá podle rovnice 3.9 [1]. Následně se pro každé slovo spočítá průměrná (podle vztahu 3.10) nebo

maximální (podle vztahu 3.11) vzájemná informace přes všechny třídy. Relevance slova w vzhledem k třídě i se potom určí vzhledem k průměrné nebo maximální vzájemné informaci pro dané slovo.

$$M_i(w) = \log \frac{F(w) \cdot p_i(w)}{F(w) \cdot P_i} = \log \frac{p_i(w)}{P_i} \tag{3.9}$$

$$M_{avg}(w) = \sum_{i=1}^{k} P_i \cdot M_i(w) \tag{3.10}$$

$$M_{max}(w) = max_i\{M_i(w)\} \tag{3.11}$$

3.4.4 Pearsonův chí-kvadrát test

Pearsonův chí-kvadrát test (dále χ^2 test), podobně jako metoda vzájemné informace, počítá korelaci mezi termy a třídami. Nechť n je počet dokumentů v datové sadě, $p_i(w)$ je podmíněná pravděpodobnost, že dokument patří do třídy i, pokud obsahuje slovo w, P_i je celkový podíl dokumentů obsahující třídu i a $F(w)$ je celkový podíl dokumentů obsahující slovo w. χ^2 test se spočítá podle rovnice 3.12 [1]. Podobně jako u metody vzájemné informace porovnáme vypočítanou hodnotu s průměrnou (3.13) nebo maximální (3.14) hodnotou χ^2 pro slovo w.

$$\chi^2(w) = \frac{n \cdot F(w)^2 \cdot (p_i(w) - P_i)^2}{F(w) \cdot (1 - F(w)) * (P_i * (1 - P_i))} \tag{3.12}$$

$$M_{avg}(w) = \sum_{i=1}^{k} P_i \cdot \chi_i^2(w) \tag{3.13}$$

$$M_{max}(w) = max_i\{\chi_i^2(w)\} \tag{3.14}$$

Kapitola 4

Klasifikátory

Tato kapitola popisuje vybrané klasifikační algoritmy. Volba klasifikátoru má významný vliv na výslednou přesnost klasifikace, a proto je důležité, aby byl zvolen optimální. V této práci se budu zabývat třemi klasifikátory – SVM (support vector machines), Bayesovským klasifikátorem a stromovým klasifikátorem.

4.1 Support vector machines

Support vector machines (dále SVM) je jednou z nejpoužívanějších metod pro klasifikaci textu. To je důsledkem vysoké dimenzionality, která je charakteristická pro textová data. Jen málo příznaků je irelevantních, ale často jsou vzájemně závislé. Jednou z výhod SVM je právě robustnost vzhledem k vysokému počtu příznaků [1].

4.1.1 Princip

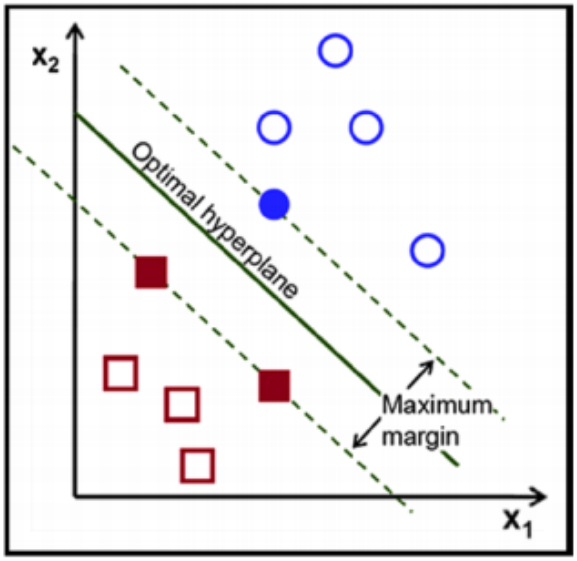
SVM je založeno na principu minimalizace strukturálních rizik (structural risk minimalization). Snažíme se nalézt takovou hypotézu h, pro kterou můžeme garantovat co nejnižší skutečnou chybu (true error). Skutečná chyba je potom pravděpodobnost, že hypotéza h bude chybná na náhodně vybraném vzorku z testovací množiny dat [4].

SVM je v základní verzi lineární klasifikátor, který rozdělí daný prostor nadrovinami (hyperplanes), jež co nejlépe oddělují třídy, které chceme klasifikovat. Nadrovinu je možné vyjádřit jako $w * x + b = 0$, kde x je vektor příznaků, w je vektor vah a b je posun. Na obrázku 4.1 čárkované linie znázorňují hranice, ve kterých je možné posunovat nadrovinu (plná čára) a přitom se vyhnout chybné klasifikaci dat. SVM se snaží tuto nadrovinu umístit tak, aby byla její vzdálenost k hraničním liniím maximální a najít tak optimální nadrovinu [16].

Prvky trénovací sady, které leží na čárkovaných liniích, se nazývají podpůrné vektory (support vectors). Pouze tyto podpůrné vektory jsou použity pro klasifikaci. Přesnost klasifikace tedy neovlivní, pokud odstraníme ostatní body, které nejsou pomocnými vektory, a výsledný model je reprezentován jen malým zlomkem původní trénovací sady [16].

4.1.2 Jádrové funkce a problém lineární separovatelnosti

Může se stát, že jednotlivé třídy nejsou lineárně separovatelné. K řešení tohoto problému se používají dva přístupy – mapování do vícedimenzionálního prostoru a metoda „soft margin hyperplanes“. V praxi se oba přístupy kombinují.

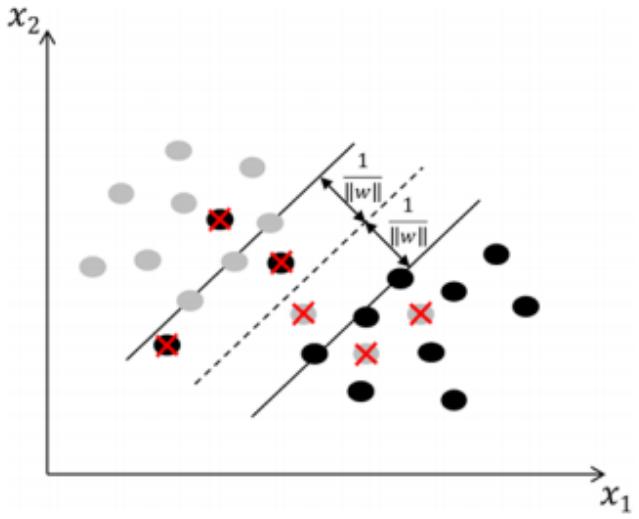


Obrázek 4.1: Oddělující nadrovina [2]

Jádrová funkce (anglicky kernel function) je funkce, která umí namapovat vzorky z datové sady do vícedimenzionálního prostoru zvaného jádrový prostor (kernel space), kde už problém bude lineárně separovatelný. Pozitivním atributem této metody je fakt, že převod do jádrového prostoru je rychlý. Byla vytvořena řada jádrových funkcí, kde mezi nejznámější patří lineární, Gaussovská, polynomiální a hyperbolický tangens. V oblasti textové klasifikace se nejčastěji využívá lineární jádrová funkce, protože textová data mají obvykle vysoký počet příznaků a je tedy pravděpodobné, že data budou lineárně separovatelná v daném vysoce dimenzionálním prostoru. Například mějme jednodimenzionální prostor příznaků s body $\{-1, 1, -2, 2\}$, kde první dva body patří do třídy A a druhé dva do třídy B. V tomto jednorozměrném prostoru nejsou lineárně separovatelné. Pokud ale transformujeme tyto body do dvojrozměrného prostoru funkcí $f(x) = (x, x^2)$, dostaneme body $[-1, 1], [1, 1], [-2, 4], [2, 4]$, které už je možné rozdělit například přímkou $y = 2.5$.

Druhým řešením je metoda, kterou představil Vapnik v roce 1995 [4]. Základní varianta SVM popsaná výše se snaží najít takovou nadrovinu, která umožní klasifikovat všechny vzorky trénovací množiny správně (hard margin). Problém je, že taková rovina nemusí obecně existovat. Je možné to řešit tím, že se umožní některé vzorky nesprávně klasifikovat. Metoda se potom snaží minimalizovat cílovou funkci a tak je možno najít nadrovinu, která sice neoddělí perfektně všechny vzorky dvou tříd, ale najde takovou nadrovinu, která vyústí v co nejmenší chybu (co nejméně vzorků na špatné straně dělící nadroviny). Princip této metody je vidět na obrázku 4.2. Obecný tvar cílové funkce představuje rovnice 4.1, kde ω je vektor vah, ξ_i je cenová funkce vzorku i, N je počet vzorků, b je posun, x_i je příznakový vektor vzorku i, y_i je třída vzorku i a C je regularizační parametr.

$$\begin{aligned} \gamma(\omega, b, \xi) &= \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{N}\xi_i \\ \text{subject to} \quad & y_i[\omega_i^T x_i + b] \geq 1 - \xi_i, \;\; i = 1, 2, \ldots, N, \\ & \xi_i \geq 0, \;\; i = 1, 2, \ldots, N. \end{aligned} \tag{4.1}$$



Obrázek 4.2: Ilustrace soft margin klasifikace [2]

Důležité je nastavení parametru C, který má za úkol určovat míru mezi snahou mít co nejnižší chybu na trénovací sadě a snahou získat obecné výsledky, které budou platit pro obecná data. Čím je parametr C vyšší, tím více se bude klasifikátor snažit, aby se vyhnul nesprávné klasifikaci vzorků na trénovací sadě. Při příliš vysokém C hrozí, že SVM bude přeučené na trénovacích datech a bude špatně zobecňovat na dosud neviděných datech. Naopak snižování C nutí SVM najít co největší oddělovače mezi třídami a je tedy povoleno více nesprávně klasifikovaných vzorků. Příliš malé C vyústí v nízkou přesnost, protože model bude příliš obecný [2]. Na obrázku 4.3 je vidět vliv parametru C na to, jaká nadrovina bude vybrána.

4.1.3 Vícetřídní klasifikace

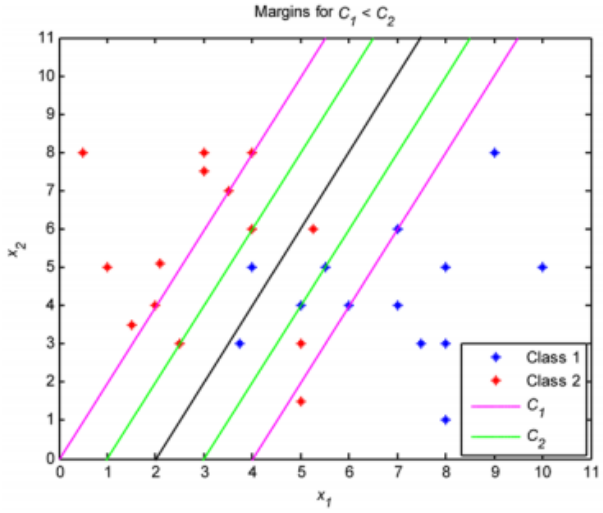
V původním modelu SVM se jedná o binární klasifikátor [2]. Umí tedy klasifikovat data pouze do dvou tříd. To ale nestačí na velkou část praktických aplikací, kde klasifikačních tříd bývá mnohem víc. Existují dva základní přístupy, jak tento problém vyřešit. V následujících rovnicích jsou použity stejné konvence jako u rovnice 4.1.

- Použije se více binárních klasifikátorů

  One-against-all mějme c tříd. Zkonstruujme c binárních SVM klasifikátorů. Každý z těchto klasifikátorů rozlišuje jednu třídu od všech ostatních, což redukuje problém na binární klasifikaci. Je zde c rozhodovacích funkcí $\omega_1^T\varphi(x_i) + b_1; ...; \omega_c^T\varphi(x_i) + b_c$, kde φ je jádrová funkce. Klasifikovaný vzorek je přiřazen třídě v případě, že klasifikátor pro danou třídu vzorek přijal a všechny ostatní je odmítly. Vzorky, které přijalo více klasifikátorů, budou přiřazeny do třídy, jehož rozhodovací funkce dává nejvyšší výstup v absolutní hodnotě podle vzorce 4.2 pro vzorek x. Složitost je $O(c * N^3)$

  $$\text{class of x} \equiv \arg\max_{i=1,...,c}(\omega_i^T\varphi(x_i) + b_i) \tag{4.2}$$

  One-against-one mějme c tříd. Zkonstruujme $c * (c-1)/2$ binárních SVM klasifikátorů. Každý z nich rozhoduje mezi dvěma třídami a má jeden hlas, který



Obrázek 4.3: Ukázka vlivu parametru C [2]

přiřadí jedné své třídě. O každé třídě hlasuje $c-1$ klasifikátorů. Výsledná třída se potom určí podle třídy s nejvyšším počtem hlasů. Složitost je $O(4 \cdot (c-1) \cdot N \cdot \frac{3}{(c \cdot 2)})$

- Zkonstruujeme cílovou funkci, která problém optimalizuje v jednom kroku. Cílová funkce vytvoří c dvoutřídních pravidel a c rozhodovacích funkcí, které řeší následující podmínky [2]:

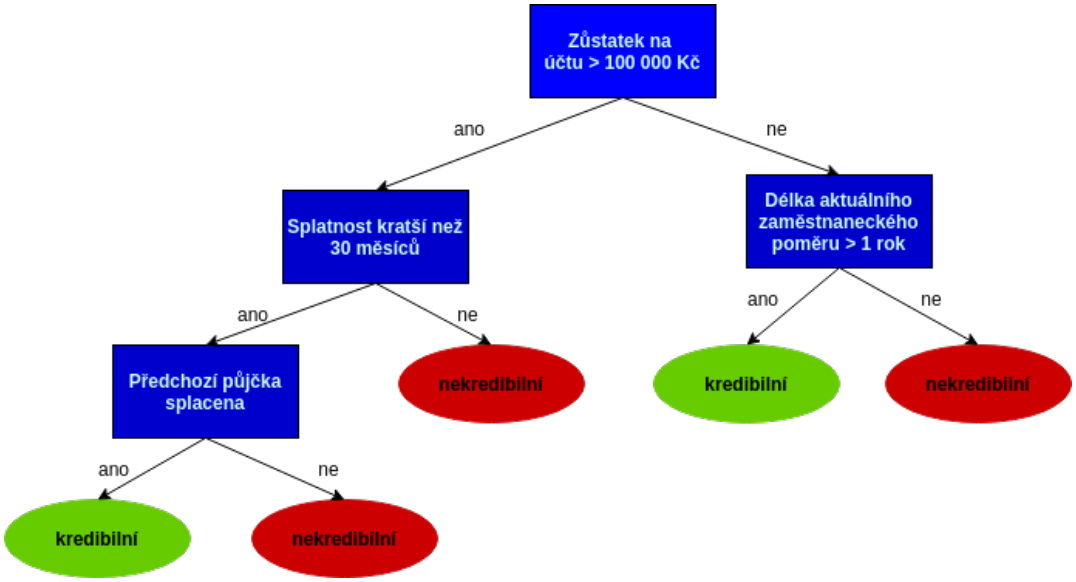
$$\omega_{y_i}^T \varphi(x_i) + b_{y_i} \geq \omega_m^T \varphi(x_i) + b_m + 2 - \xi_m^i, \xi_m^i > 0; \tag{4.3}$$

4.1.4 Škálovatelnost

Problémem SVM je špatná škálovatelnost na velkých datových sadách. Konvergence na trénovacích datech je pomalá, a to hlavně u nelineárních jádrových funkcí. Dalším problémem je, že počet podpůrných vektorů nejde dopředu určit a někdy mohou podpůrné vektory tvořit až polovinu datové sady. Potom i rychlost klasifikace bývá pomalá [2]. Existuje řada metod pro urychlení SVM. Platt [18] vytvořil metodu SMO (sequential minimal optimization), která rozděluje optimalizační problém SVM na dva kvadratické problémy (Quadratic program problem) a umožňuje trénovat se složitostí lineární nebo kvadratickou v závislosti na velikosti datové sady. Dále existují metody pro paralelizaci SVM jak na procesoru, tak na GPU.

4.2 Rozhodovací stromy

Rozhodovací strom je ve své podstatě hierarchická dekompozice trénovacího prostoru, kde dělení probíhá na základě predikátů nebo podmínek na hodnotu atributu. U testových dat se potom obvykle jedná o podmínku přítomnosti nebo absence určitého slova či jiného příznaku [1]. Výhodou těchto klasifikátorů je, že umí pracovat s různými typy atributů, ať už to jsou atributy numerické, ordinální, kategorické nebo nominální. Klasifikační stromy jsou běžně používány v oblastech jako jsou finance, marketing, medicína. Výsledky stromových klasifikátorů jsou jednoduše interpretovatelné.



Obrázek 4.4: Příklad rozhodovacího stromu

4.2.1 Princip

Rozhodovací stromy mají dva typy uzlů: interní (také zván testovací) uzel, který na základě hodnot atributů dělí strom do více větví, a listový, ve kterém jsou odpovídající třídy, jež jsou výsledkem klasifikace. Na obrázku 4.4 je příklad jednoduchého rozhodovacího stromu pro určení kredibility klientů banky. Zde jsou interní uzly modré a listové uzly zelené a červené. V kořenovém uzlu se data dělí na základě podmínky *Zůstatek na účtu je vyšší než 100 000 Kč*. Klasifikátor se podívá na odpovídající atribut v datech a vyhodnotí podmínku. Na základě této podmínky se data rozdělí na dvě části a znovu se aplikují na obě poloviny další podmínky. To se děje do té doby, dokud není dosažena určitá hranice. Často používané podmínky jsou: dosažení maximální výšky stromu, dosažení minimálního počtu záznamů v listových uzlech nebo pokud data v uzlu nedosáhnou určeného stupně čistoty (je zde vysoký podíl záznamů z jedné třídy). Výsledná třída v listovém uzlu se potom určí obvykle jako majoritní třída vzorků, které se v listovém uzlu nacházejí.

Klasifikace potom probíhá podobným způsobem. Načítají se vzorky z testovací sady a postupuje se od kořene, aplikují se podmínky a na základě výsledku podmínky v uzlu se postupuje směrem k listovým uzlům. Listový uzel, ve kterém algoritmus pro daný vzorek skončí, určí výslednou klasifikační třídu.

Pro rozdělení uzlu je možné použít řadu různých podmínek. Rozlišují se tři základní typy [1]:

- Rozdělení podle jednoho atributu: z hlediska textové klasifikace se jedná o rozdělení na základě přítomnosti nebo absence určitého příznaku (slova, fráze). V každé úrovni se vybere ten atribut, který nám umožní co nejlépe diskriminovat data na dvě části. Pro určení, jaký atribut to bude, se používají metody jako Gini index a informační přínos, které jsou popsány v kapitole 3.4 o výběru příznaků.

- Rozdělení více atributů podle metriky podobnosti: zde se pracuje s dokumenty nebo s daty vytvořenými z dokumentů (shluky slov) a počítá se podobnost mezi dokumenty pro rozhodnutí, kde provést rozdělení. Dokumenty se seřadí podle podobnosti a podle

dané hranice se rozdělí. Jako referenční bod se vybere takový shluk slov, který vyústí v co nejlepší rozdělení dat.

- Rozdělení více atributů podle diskriminantu: k rozdělení je možné použít diskriminanty (např. Fisherův diskriminant). Diskriminanty stanoví směr v datech, ve kterém je nejlepší data separovat.

Stromové klasifikátory úzce souvisí s klasifikátory založenými na pravidlech a odvozování pravidel. Každá cesta z kořene do listu může být transformována na jedno pravidlo konkatenací pravidel, na které se narazí po cestě do listového uzlu [21].

Důležitá je také velikost stromu, která má velký vliv na přesnost klasifikace [21]. Obvykle jsou preferovány jednodušší stromy, protože s těmi lze snadněji pracovat. Většinou je složitost stromu měřena počtem uzlů, počtem větví, hloubkou stromu nebo počtem použitých atributů.

4.2.2 Prořezání stromu

Stejně jako u ostatních metod strojového učení může dojít k přeučení nebo podučení klasifikátoru. Přeučení u stromových klasifikátorů obvykle nastane, pokud má strom příliš mnoho uzlů vzhledem k množství dat, se kterými pracuje. Naopak podučení nastane, pokud strom obsahuje příliš málo uzlů, a je tedy příliš obecný. Dva hlavní přístupy, jak těmto problémům předcházet, jsou: nerozdělovat uzel, pokud rozdělení není užitečné podle určité metriky (povolit pouze statisticky významné rozdělení), druhým přístupem je prořezání stromu [21].

Při použití striktních pravidel pro zastavení konstrukce stromu dostáváme menší stromy a hrozí podučení. Naopak při volných pravidlech budou výsledkem velké komplexní stromy a hrozí přeučení. Jednou z metod, jak tomu předejít, je nechat rozhodovací strom se přeučit a potom odstranit ty větve, které mají malou rozlišovací schopnost (prořezání). Další motivací pro prořezání stromu může být získání jednoduššího stromu, který je jednodušeji interpretovatelný. Metody pro prořezávání obvykle procházejí stromem shora-dolů nebo zdola-nahoru a uzel je zrušen, pokud tato operace zlepší určité kritérium. Příkladem prořezávací metody může být *Cost complexity pruning* [3], kde se zkonstruuje k stromů T_0 až T_k, kde T_0 je původní neprořezaný strom a T_k je strom pouze s kořenovým uzlem. Jeden z těchto stromů je vybrán na základě metriky odhadu chyby [21].

4.2.3 Rozhodovací lesy

Myšlenkou rozhodovacích lesů je, že kombinací více modelů, kde každý z nich řeší stejnou úlohu, je možné dosáhnout lepších výsledků, než kdybychom použili pouze jeden model. Těmto metodám se říká souborné metody (ensemble methods). Obvykle každá taková metoda obsahuje čtyři základní bloky:

- trénovací data,
- indukční algoritmus: algoritmus, který na základě dat na vstupu vytvoří klasifikátor, jenž reprezentuje vztah mezi vstupními atributy a výstupní třídou,
- generátor: je zodpovědný za vygenerování jednotlivých klasifikátorů,
- kombinátor: má za úkol zkombinovat výsledky jednotlivých stromů do výsledného rozhodnutí.

Základním algoritmem je vytvoření n stromů a každý dostane na vstup určitou podmnožinu vstupních dat. Některé položky se mohou v určitých sadách opakovat, některé tam nemusí být vůbec. Následně každé rozhodnutí lesa je kombinací výsledků z jednotlivých stromů.

Ke kombinaci výsledků z klasifikátorů jsou dva základní přístupy:

- Váhovací metody: k jednotlivým stromům se přiřadí váhy a podle nich se spočítá výsledek. Může se jednat o majoritní hlasování, kde má každý strom stejnou váhu nebo se může váha určit na základě přesnosti daného klasifikátoru na datech.
- Meta-kombinační metody: aplikují se klasifikační algoritmy na rozhodovací stromy a snaží se určit, které natrénované stromy jsou spolehlivé a které ne. Jedna část datové sady je použita pro natrénování stromů a druhá část potom pro jejich výběr.

4.3 Naivní Bayes

Naivní Bayes (dále NB) je jedním z nejznámějších zástupců třídy generativních klasifikátorů. Modeluje distribuci dokumentů v každé třídě pomocí pravděpodobnostního modelu, který předpokládá nezávislost mezi výskytem termů v dokumentu (proto se nazývá naivní). Je obtížné zkonstruovat Bayesovský klasifikátor, který by počítal se závislostmi mezi termy, a to z důvodu vysokých nároků na výpočetní výkon a problematického zpracování malého množství dat. Určitý pokrok byl ale dosažen v oblasti, kde se uvažuje jen částečná závislost mezi termy [14]. Dvě skupiny modelů se běžně používají pro NB klasifikaci. Oba neberou ohled na pořadí termů v dokumentu (podobně jako u BOW modelu).

- Bernoulliho model: zde se jako příznaky využívá přítomnost nebo absence slov v dokumentu. Tento model tedy používá binární příznaky (popsané v kapitole 3.2).
- Multinomický model (MNB): U tohoto modelu se zaznamenává i frekvence jednotlivých slov. Dokument může být modelován jako vzorek z multinomického rozdělení.

Naivní Bayes spočítá pravděpodobnost toho, že daný dokument patří do určité třídy a poté je dokument přiřazen do třídy s nejvyšší pravděpodobností.

Podle Kontose [13] je Bernoulliho model lepší při menších slovnících, naopak multinomický model má lepší výsledky při větších slovnících a skoro vždy dává lepší výsledky než Bernoulliho model, pokud jsou pro oba vybrány optimální velikosti slovníků.

Manning a Wang [27] provedli srovnání SVM a multinomického Bayesova klasifikátoru a došli k závěru, že MNB je lepší pro krátké dokumenty, zatímco SVM má lepší výsledky na delších.

4.3.1 Bernoulliho model

Mějme slovník termů $V = \{t_i, ..., t_n\}$, dokument $Q = \{t_{i_1}, ..., t_{i_m}\}$ a třídy $1...k$. Cílem je potom modelovat pravděpodobnost, že dokument T patří do třídy i, pokud obsahuje termy $Q = \{t_{i_1}, ..., t_{i_m}\}$.

Mějme množinu T, což je množina termů, kde pravděpodobnostní rozložení jednotlivých termů odpovídá některé z tříd 1 až k. Bernoulliho model se snaží spočítat podmíněnou pravděpodobnost, že T patřilo do třídy i, pokud po vzorkování T jsme dostali množinu Q. Apriorní pravděpodobnost výběru třídy i je rovna jejímu podílu z celkového počtu dokumentů v datové sadě.

Musíme spočítat následující pravděpodobnosti:

- Jaká je původní pravděpodobnost, že množina T je vzorek z rozložení termů podle třídy i? Tato pravděpodobnost je značena $P(C^T = i)$.
- Pokud bychom vzorkovali L termů z rozložení termů třídy i (s opakováním), jaká je pak pravděpodobnost, že vzorkovaná množina T je rovna množině Q? Tato pravděpodobnost se značí $P(T = Q|C^T = i)$

$$P(C_T = i|T = Q) = \frac{P(C^T = i) \cdot P(T = Q|C^T = i)}{P(T = Q)} = \\ = \frac{P(C^T = i) \cdot \prod_{t_j \in Q} P(t_j \in T|C^T = i) \cdot \prod_{t_j \notin Q}(1 - P(t_j \in T|C^T = i))}{P(T = Q)} \tag{4.4}$$

To lze zapsat pomocí Bayesovského pravidla 4.4 [1]. Výsledná třída Q se vybere jako třída s nejvyšší pravděpodobností. Jmenovatel z pravidla je možné u klasifikace zanedbat, protože přesný výpočet pravděpodobnosti není důležitý. Jde o identifikaci třídy s nejvyšší pravděpodobností. Potom dostaneme rovnici 4.5, kde $P(C^T = i)$ je pravděpodobnost dokumentů, které patří do třídy i, $P(t_j \in T|C^T = i)$ je podíl dokumentů v i-té třídě, která obsahuje term t_j a $P(t_j \in T)$ je podíl dokumentů v datové sadě obsahující term t_j. V praxi se používá Laplaceho vyhlazování, kde se k vypočítaným frekvencím přidávají malé hodnoty, aby se zabránilo nulové pravděpodobnosti u řídce se vyskytujících termů.

$$i' = \arg\max_i P(C^T = i|T = Q) \tag{4.5}$$
$$= argmax_i P(C^T = i) \cdot P(C^T = i) \cdot \prod_{t_j \in Q} P(t_j \in T|C^T = i) \cdot \prod_{t_j \notin Q}(1 - P(t_j \in T|C^T = i))$$

4.3.2 Multinomický model

Mějme slovník termů $V = \{t_i, ..., t_n\}$ a dokument $Q = \{t_{i_1}, ..., t_{i_m}\}$ s příslušnými frekvencemi termů $F = \{F_{i_1} \dots F_{i_m}\}$. Celkový počet termů v dokumentu je L, kde $L = \sum_{j=1}^{m} F_{i_j}$. Termy Q s příslušnými frekvencemi F budeme značit $[Q, F]$. Cílem je potom modelovat pravděpodobnost, že dokument T patří do třídy i, pokud obsahuje prvky $[Q, F]$.

Mějme množinu T s L termy, která byla vytvořena vzorkováním s návratem z množiny termů, jenž odpovídá pravděpodobnostnímu rozložení jedné z tříd 1 až k. Multinomický model se snaží spočítat podmíněnou pravděpodobnost, že T patřilo do třídy i, pokud po vzorkování T jsme dostali množinu Q s frekvencemi F. Apriorní pravděpodobnost výběru třídy i je rovna jejímu podílu z celkového počtu dokumentů v datové sadě.

Výše zmíněná pravděpodobnost výběru třídy i s dokumentem T je zapisována jako $P(C^T = i|T = [Q, F])$. Dalším obvyklým zjednodušením je, že třída není závislá na délce dokumentu. Pro spočítání pravděpodobnosti potřebujeme odhadnout dvě hodnoty:

- Jaká je původní pravděpodobnost, že množina T je vzorek z rozložení termů podle třídy i? Tato pravděpodobnost je značena $P(C^T = i)$.
- Pokud bychom vzorkovali L termů z rozložení termů třídy i (s opakováním), jaká je pravděpodobnost, že naše vzorkovaná množina T je množina Q s příslušnými frekvencemi F? Tato pravděpodobnost se značí $P(T = [Q, F]|C^T = i)$.

Bayesovské pravidlo se potom zapíše podle rovnice 4.6 [1].

$$P(T = [Q, F]|C^T = i) \tag{4.6}$$
$$= \frac{P(C^T = i) \cdot P(T = [Q, F]|C^T = i)}{P(T = [Q, F])} \quad \propto P(C^T = i) \cdot P(T = [Q, F]|C^T = i$$

Hodnotu jmenovatele $P(T = [Q, F]|C^T = i)$ lze odhadnout podle podílu dokumentů patřící do třídy i v datové sadě. Pokud uvážíme sekvenční pořadí L vzorků, tak počet možných způsobů jak vzorkovat termy, abychom dostali $[Q, F]$, je dána jako $\frac{L!}{\prod_{i=1} mF_i!}$. Pravděpodobnost každé z těchto sekvencí je potom $\prod_{t_j \in Q} P(t_j \in T|C_T = i)^{F_j}$. Dostaneme tedy rovnici 4.7 [1]. Ta se dosadí do rovnice 4.6 a vypočítáme třídu s nejvyšší pravděpodobností.

$$P(T = [Q, F]|C^T = i) = \frac{L!}{\prod_{i=1} mF_i!} \cdot \prod_{t_j \in Q} P(t_j \in T|C_T = i)^{F_j} \tag{4.7}$$

Kapitola 5

Příprava testovacích dat

Od firmy Mavenir jsem dostal k dispozici agregovaná data SMS kampaní. Data jsou rozdělena do souborů ve formátu csv podle regionů, ve kterých byla získána. Na každém řádku je záznam z jedné kampaně, tak jak byla rozpoznána detekčním systémem. Každý záznam obsahuje pro danou kampaň následující údaje:

- počet zpráv v kampani,
- počet mobilních čísel rozesílajících zprávy v kampani,
- náhodně vybraný reprezentant z textů rozeslaných během kampaně.

Pro účely klasifikace je důležitá zejména třetí položka se samotným textem. Jedná se o dostatečnou informaci pro reprezentaci celé kampaně, protože zprávy v rámci jedné kampaně si bývají velmi podobné.

region	jazyk	počet vzorků
Argentina	španělština	14 621
Brazílie	portugalština	7 207
Německo	němčina	9 838
Indonésie	indonéština, angličtina	12 368

Tabulka 5.1: Přehled datových sad

K dispozici byla pouze textová data bez jakýchkoliv přiřazených kategorií či tříd. Vzhledem k tomu, že většina metod strojového učení využívá modelu učení s učitelem jsou potřeba pro trénování klasifikačního modelu anotovaná data s přiřazenými třídami. Bylo nutné určit, jaké jsou vhodné typy kategorií a následně část dat manuálně klasifikovat. Manuální klasifikace všech dat je nereálná, protože dodané soubory s daty měly tisíce záznamů o kampaních. V tabulce 5.1 je přehled datových sad včetně počtu zpráv, které každá obsahuje. Cílem bylo ověřit na anotované podmnožině celé datové sady přesnost a spolehlivost vytvořeného klasifikačního modelu. Následně bude možné strojově anotovat celou datovou sadu.

Prvním krokem je určení typu kategorií. Kategorie by měly pokrývat sémantický význam textu, ale otázkou je, jak specifické by měly kategorie být. Nakonec jsem se rozhodl, použít dva typy kategorií s různou mírou specifičnosti. Obecnější skupiny kategorií by měly texty rozdělit na základě oborů či oblastí, kterých se texty týkají. Druhý typ by měl texty dělit

Instala DIRECTV HD a CERO Pesos! Manda OK p/ info

Te interesa un Ford 0km en cuotas sin interes? Responde OK para info

Estimado cliente,su estado de mora con CREDIAL ya supera los 60 dias. Comuniquese dentro de las proximas 48hs al (XXX)XXXXXXXX.Evite acciones futuras

Obrázek 5.1: Příklad spamových SMS zpráv

na základě konkrétního subjektu, jenž kampaň rozeslal. První tedy bude obecnější, zatímco druhý bude specifičtější. Každopádně se ale nedá přímo říci, že každá kategorie z druhého rozdělení je podmnožinou některé kategorie z prvního rozdělení.

Dalším problémem je, že vytvořené kategorie se vztahují hlavně ke kampaním z jedné oblasti a obvykle nejsou aplikovatelné na kampaně z oblasti jiné. Důvodem je to, že kampaně se značně liší podle regionů, zejména z hlediska obsahu a oborů subjektů, které kampaně rozesílají. Proto je obtížné vytvořit obecný systém kategorií, který bude užitečný a aplikovatelný pro všechny oblasti. V následujícím textu budou použity příklady kategorií pro data kampaní z Argentiny. Jak vypadají typické zprávy z této datové sady, je možné vidět na obrázku 5.1. První informuje, že instalace kabelové televize od DirecTV je zdarma. Druhá zpráva nabízí koupi Fordu na splátky bez úroků. Třetí vyzývá klienta k zaplacení pohledávky.

5.1 Kategorie

V následující sekci podrobněji popíšu navržené kategorie tříd a ukážu, jak jejich použití dopadlo na argentinské datové sadě.

5.1.1 Kategorie podle oboru

Cílem této skupiny kategorií je rozdělit data podle obecného oboru, se kterým text souvisí. Po analýze obsahu tisícovky textů jsem navrhl následující kategorie:

debt-warn jde o výzvy k zaplacení dluhu, případně varování před následky v případě nezaplacení,

tv-ad reklama na televizní poskytovatele,

info zpráva informuje uživatele o nějaké události, např. dodání zboží,

payment upomínka k zaplacení platby,

general-ad reklama, která nespadá do žádné jiné kategorie,

car-ad automobilová reklama,

loan nabídka peněžní půjčky,

health-ad reklama v oblasti zdravotnictví,

operator-ad reklama na mobilního operátora,

nonsense zpráva bez nějakého zjevného významu,

automatic automatické odpovědi, například registrační kódy, obnova ztraceného hesla. . . ,

embargo oznámení o uvalení soudního embarga, obvykle z důvodu dluhů,

promo oznámení o konané promoční akci,

event oznámení o konání nějaké akce,

housing-ad reklama v oblasti bydlení,

app-ad reklama na mobilní aplikace,

other ostatní zprávy nespadající do žádné z ostatních kategorií,

contact snaha spojit se s uživatelem, obvykle žádost, aby zavolal na dané číslo bez zmínění účelu.

Z rozdělení je vidět, že zde jsou dvě hlavní oblasti. Jednak reklamy na různé produkty a potom byznys s půjčkami a dluhy. Na obrázku 5.2 jsou počty zpráv ve výše zmíněných třídách. Je vidět, že kategorie jsou nevyvážené. Kategorie *debt-warn* je téměř třikrát objemnější než druhá největší třída a tvoří více než čtvrtinu všech zpráv ve zkoumaném vzorku. Následně zde je skupina deseti tříd s počty zpráv v desítkách. Třetí skupinu potom tvoří minoritní třídy s počty zpráv do 20. Zprávy na obrázku 5.1 by byly zařazeny do kategorií *tv-ad*, *car-ad* a *debt-warn*.

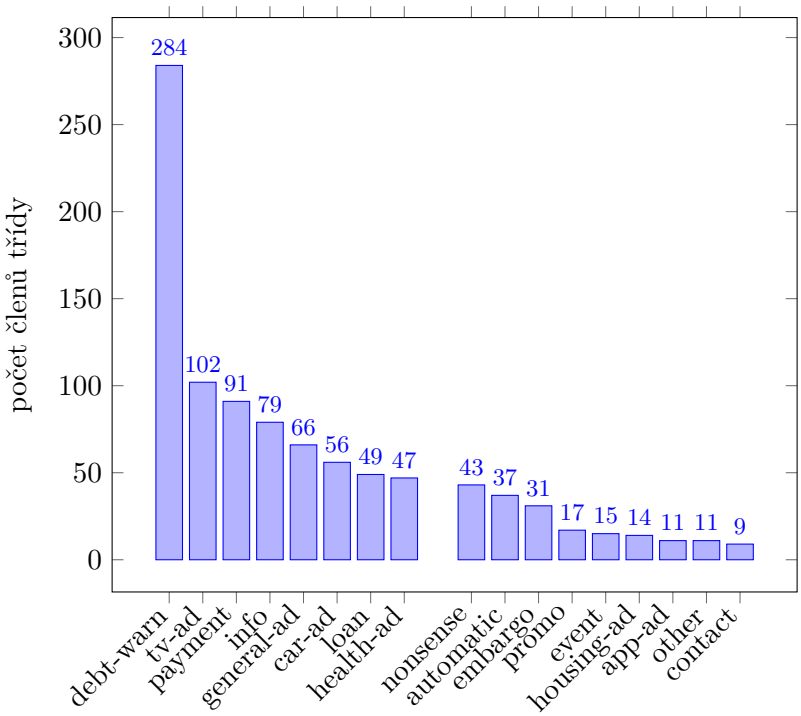
5.1.2 Kategorie podle subjektů

Tento typ kategorií je více specifický. Cílem je určit subjekt (obvykle firmu), který za kampaní stojí. Tedy například pokud se jedná o reklamu, tak to bude firma, jejíž produkt reklama propaguje. Tento typ kategorií je užitečný, protože to umožňuje identifikovat subjekty, které kampaně zasílají a následně zákazníci firmy Mavenir se s nimi mohou domluvit na podmínkách zasílání těchto kampaní.

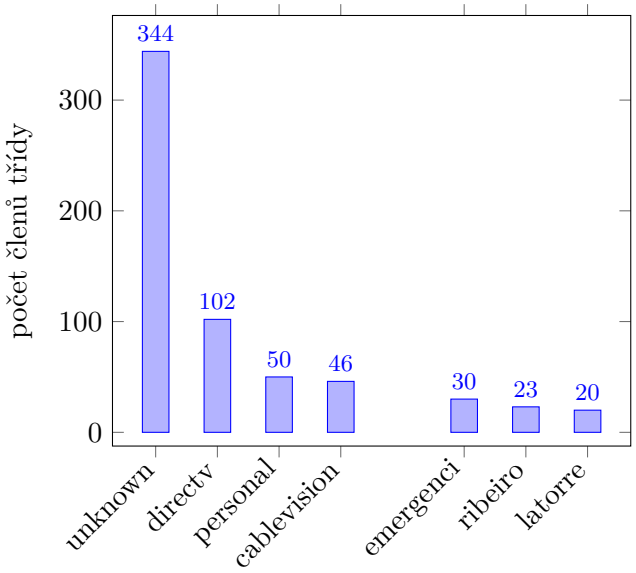
Vzhledem k nižší obecnosti této kategorie výrazně narostl počet tříd. Je zde 75 tříd. Popíši tedy jen jejich významné charakteristiky. Na obrázku 5.3 jsou zobrazeny počty sedmi nejfrekventovanějších tříd. Jedná se o názvy firem. DirectV a Cablevision jsou televizní poskytovatelé, Personal je telefonní operátor, Ribeiro a Latorre podnikají v oblasti peněžních půjček. Míra nevyváženosti je zde ještě vyšší než u předchozí kategorie. Nejvíce početná je třída zpráv, u kterých se nepodařilo zjistit odpovědný subjekt (označená v grafu jako *none*). Jedná se o více než třetinu všech zpráv (344). Následuje 5 kategorií s několika desítkami zpráv. Dalších pět je v rozmezí 10–20. Ostatní, což je kolem 60 kategorií, jsou méně početné s méně než 10 zprávami. Pokud se subjekt vyskytl pouze v jednom textu, tak byl přeřazen do kategorie neznámých subjektů.

Problematický je zde zejména vysoký počet tříd s velmi malým počtem vzorků. V tomto případě je pro klasifikátor velmi složité naučit se určovat tyto třídy. Zejména třídy s méně než pěti vzorky mají nízkou šanci být správně klasifikovány. Také je možné, že při náhodném rozdělení do testovací a trénovací sady nebude v trénovací sadě žádný vzorek této třídy.

Zprávy na obrázku 5.1 by byly zařazeny do kategorií *directv*, *ford* a *credial*.



Obrázek 5.2: Počty zpráv v navržených třídách podle oboru



Obrázek 5.3: Počty zpráv v navržených třídách podle subjektu

5.1.3 Kombinace

Dvě výše zmíněné kategorie tříd je možné kombinovat. Zejména pro kategorie podle subjektů se často může stát, že známých subjektů je pouze minimum a zbytek potom spadá do jedné sjednocující kategorie neznámých subjektů. To ale pro cíl úlohy není moc užitečné. Proto je možné známé subjekty zachovat a ty neznámé rozdělit do kategorií podle oborů.

5.2 Postup anotace

Výsledná přesnost klasifikace nezáleží jen na zvolených algoritmech a metodách, ale také na vstupních datech. Pokud je obtížné v datech najít vzory, bude přesnost nízká. Proto je velmi důležité vhodně zvolit množinu tříd, do kterých se budou datové prvky přiřazovat. V následující sekci se pokusím shrnout, jakým způsobem jsem prováděl anotaci a jaké postupy návrhu tříd a kategorií tříd dávaly dobré výsledky.

Průběh anotace by měl být víceprůchodový. Pokud datová sada projde jen jednou, je pravděpodobné, že výsledné třídy nebudou konzistentní. Na začátku není známo, jak data vypadají a jaké třídy budou potřeba. Při prvním průchodu je hlavně důležité seznámit se s charakterem dat a určit, jaké třídy má smysl uvažovat. I při prvním průchodu se přiřazují třídy, ale spíše orientačně. Není nutné všem prvkům nějakou třídu přiřadit, je možné některé nechat na později. Při druhém průchodu je již známo, jak data vypadají a vyvstávají dva hlavní cíle. Prvním je opravit nekonzistence z prvního průchodu. Pokud například v polovině průchodu usoudí, že by bylo vhodné zavést třídu X pro určitou skupinu vzorků, tak je nutné všechny vzorky v první polovině, které by měly patřit do dané třídy, patřičně upravit. Druhým cílem je doplnění vzorků, které byly vynechány při prvním průchodu. Jelikož už je v této fázi známé, jak data vypadají, tak by mělo být jednodušší určit, do jaké třídy mají patřit. Následně je možné provést více průchodů, kde se znovu rozdělení do tříd vylepšuje, ale alespoň dva průchody jsou vhodné.

Při návrhu tříd je vhodné vzít v potaz několik faktorů. Nejdůležitější je velikost třídy, která by neměla být nižší než 5–10 vzorků a ideálně by v datové sadě neměla být ani třída, která je výrazně větší než ostatní. Dalším faktorem je obsahová vazba textu na třídu. Texty dané třídy by měly obsahovat termy, které danou třídu identifikují oproti ostatním. Může se jednat například o název firmy u kategorií podle subjektu. Pokud třída žádná taková slova neobsahuje, tak je pravděpodobné, že se bude plést s příbuznými třídami a je vhodné zvážit, zda by se neměla sloučit s jinou příbuznou třídou.

Pro vzorky, u kterých je jejich zařazení obtížné, je možné vytvořit sběrnou třídu, kam se budou umisťovat vzorky, které nepatří do žádné jiné třídy, ale současně v datové sadě není dostatek podobných vzorků, aby to umožnilo vytvořit samostatnou třídu. Tato třída bude mít nízkou výslednou přesnost, ale výhodou je, že obtížně zařaditelné vzorky nekontaminují ostatní třídy, které díky tomu budou mít vyšší přesnost. Při zpracování výsledků je možné sběrnou třídu odstranit nebo ji nebrat v úvahu.

5.3 Charakteristika datových sad

V tabulce 5.2 je možné vidět počty tříd v datových sadách podle kategorií. Počty tříd v kategoriích podle oboru jsou značně podobné a pohybují se okolo 15. U kategorií podle subjektu bývá mnohem vyšší počet tříd, řádově desítky. Je to způsobeno tím, že tato kategorie je méně obecná a unikátních firem je v datové sadě velké množství. Odlišná je indonéská

	kategorie podle oboru	kategorie podle subjektu
argentinská	18	73
brazilská	16	51
indonéská	16	23
německá	12	-

Tabulka 5.2: Počet tříd v anotovaných datových sadách

sada, kde je počet tříd výrazně nižší. Důvodem je, že na této datové sadě jsem vyzkoušel kombinační způsob anotace (malé třídy jsou sjednoceny do jedné obecnější).

Argentinská datová sada je největší z hledisku počtu vzorků. V této sadě dominuje zejména reklama, a to hlavně na půjčky. Většina zpráv se týká vymáhání dluhů od dlužníků, výzev k zaplacení pohledávek nebo nabídek nových půjček. Také je častá reklama na televizní poskytovatele a mobilní operátory. Je zde minimum zpráv osobního charakteru. Část zpráv jsou automaticky vygenerované kódy a upozornění. Německá datová sada naopak obsahuje mnohem vyšší podíl zpráv osobního rázu a automaticky generovaných zpráv a je zde naopak je nižší podíl reklamních sdělení. Z osobních SMS zpráv dominují zejména pozvánky na různé akce, případně informativní SMS o různých událostech. Častá jsou upozornění na připravené objednávky, jako jsou zboží či taxi služby. Brazilská datová sada je zase spíše podobná argentinské, také je zde vysoký podíl zpráv týkajících se půjček, plateb a dluhů. Zbytek zpráv tvoří převážně reklama. Indonéská sada je více vyvážená a objevují se zde tři hlavní kategorie: bankovní zprávy, reklama na hazard a obecné reklamní zprávy.

Kapitola 6

Program

V následující kapitole je popsána programová část diplomové práce. První část se zabývá návrhem programu a druhá část jeho implementací.

6.1 Návrh

Cílem je vytvořit program, který umí klasifikovat texty do kategorií na základě sady trénovacích dat.

Jelikož texty budou tvořeny SMS zprávami, lze jejich délku očekávat mezi 50–200 znaky, obvykle s jednou až třemi větami. Mezi SMS zprávami budou jak zprávy osobního charakteru tak (ve většině případů) reklamní kampaně, které různé subjekty zasílají obvykle za účelem šíření reklamy. U spamových zpráv lze očekávat vyšší podíl URL odkazů a telefonních čísel.

6.1.1 Funkce

Hlavní funkcí programu bude predikce tříd neznámých textů. Jedním vstupem programu potom bude množina textů, u kterých kategorie není známá. K natrénování klasifikátoru jsou potřeba anotované texty, jež mají přiřazené správné třídy. Tyto texty budou tvořit druhý vstup programu. Trénování klasifikátoru obvykle trvá výrazně déle než následná predikce tříd. Proto by mělo být možné natrénovaný klasifikátor uložit do souboru a při dalším spuštění programu ho načíst. To velmi urychlí práci s programem, protože nebude nutné před každou predikcí znovu trénovat klasifikátor.

Program by měl uživateli umožňovat výběr metod a jejich parametrů. Neexistuje jedna metoda, která by byla nejlepší na všechny typy textů nebo aplikací. Proto je lepší, pokud si uživatel může vybrat jaké metody se mají použít a vyzkoušet, která je nejvhodnější. K tomu je potřeba, aby byla k dispozici zpětná vazba ohledně výsledné přesnosti současného nastavení. Další funkcí proto bude odhad přesnosti pro aktuální nastavení programu. K tomu se použije trénovací sada, pro niž se pomocí křížové validace spočítá přesnost modelu.

6.1.2 Uživatelské rozhraní

Program bude ovládán jako konzolová aplikace. Hlavním důvodem je, že primární výstup programu jsou predikované třídy, které se zapisují do souboru nebo do databáze a zde by grafické uživatelské rozhraní nijak výrazně nepomohlo. Stejně tak pro funkci statistik je vhodné výstup vypsat buď na standardní výstup nebo do souboru. Jediná výhoda grafic-

kého rozhraní by byla v intuitivnější manipulaci s nastavením programu, kde se u konzolové aplikace musí použít buď konfigurační soubor nebo argumenty při spuštění programu. Nakonec jsem zvolil, že nastavení programu bude ovládáno pomocí argumentů při spuštění programu. Povinné argumenty budou pouze dva, které zvolí primární funkci (predikce nebo odhad přesnosti) a vstup (soubor s trénovacími daty nebo databáze). U ostatních přepínačů bude potom snaha zvolit dostatečně robustní výchozí hodnoty, aby se muselo nastavení přepisovat jen výjimečně.

6.1.3 Knihovny

Algoritmy popsané v teoretické části práce jak pro extrakci a výběr příznaků nebo pro klasifikaci jsou implementovány v řadě různých knihoven. Knihovny se liší podporovanými jazyky, efektivitou implementace, množstvím implementovaných algoritmů a jednoduchostí použití. V následující sekci popisuji dvě vybrané knihovny pro strojové učení a zpracování přirozeného textu – *scikit-learn*[1] a *ML Lib*[2].

scikit-learn

Jedná se o otevřený software pod BSD licencí, který slouží jako knihovna pro strojové učení. Je založen na knihovně Scipy, což je knihovna pro vědecké výpočty. Scikit-learn je napsán v Pythonu, ale některé části, které jsou kritické pro výkon, jsou napsané v Cythonu, což je rozšíření jazyka Python, jehož cílem je přiblížit se rychlostí jazyku C.

Knihovna se skládá z šesti základních modulů:

- Classification: tento modul obsahuje desítky klasifikačních algoritmů, např. SVM, LDA, hledání nejbližších sousedů, naivní bayesovský klasifikátor, rozhodovací stromy a lesy, neuronové sítě a další.
- Regression: zde jsou regresní modely.
- Clustering: sbírka shlukovacích algoritmů, mimo jiné K-means, Affinity propagation, Mean shift, DBSCAN a další.
- Dimensionality reduction: obsahuje implementované algoritmy a metody pro výběr příznaků a redukci dimenzionality dat. Spadají sem metody výběru příznaků popsané v této práci v sekci 3.4.
- Model selection: tento modul slouží k validaci výsledků, možnosti jejich zobrazení a uložení a také algoritmy pro optimalizaci parametrů modelů.
- Preprocessing: poskytuje přístup k různým metodám předzpracování dat. Je zde zejména dobrá podpora pro předzpracování textových dat.

Jedná se o jednu z nejčastěji používaných knihoven pro strojové učení. Výhodou je velmi rozsáhlý výběr moderních algoritmů, jak pro strojové učení, tak pro předzpracování a validaci výsledků.

[1] http://scikit-learn.org/stable/
[2] https://spark.apache.org/mllib/

ML Lib

ML Lib je součástí Apache Spark, což je open source (otevřený software) framework pro distribuované výpočty. Je licencován pod Apache License 2.0. Podporuje jazyky Java, Scala, Python a R.

Scikit je velmi efektivní pokud k běhu programu stačí RAM dostupná na jednom PC. Naopak ML Lib se zaměřuje na strojové učení na velkých objemech dat, kde je nutné problém distribuovat na více výpočetních jednotek. ML Lib je proto méně efektivní na malých a středně velkých datových sadách, kde se nedá plně využít jejích předností.

Podobně jako scikit-learn obsahuje většinu běžně používaných algoritmů strojového učení.

Výběr knihovny

Firma Mavenir má v jedné datové sadě řádově statisíce datových položek. Každá položka je jedna SMS zpráva, která reprezentuje detekovanou kampaň. To při několika desítkách či stovkách příznaků na jednu kampaň znamená velikost datové sady v jednotkách gigabajtů. Tato velikost se dá výrazně zredukovat díky řídkým maticím, které jsou implementované v knihovně numpy[3]. Typický příznakový vektor textu bude mít většinu hodnot nulovou a použití řídkých matic ušetří značné množství prostoru (často více než tři čtvrtiny).

Na základě úvahy výše jsem se rozhodl použít knihovnu Scikit-learn, protože by objem dat, se kterými bude program pracovat neměl překročit kapacitu RAM a není proto potřeba používat ML Lib.

6.2 Implementace

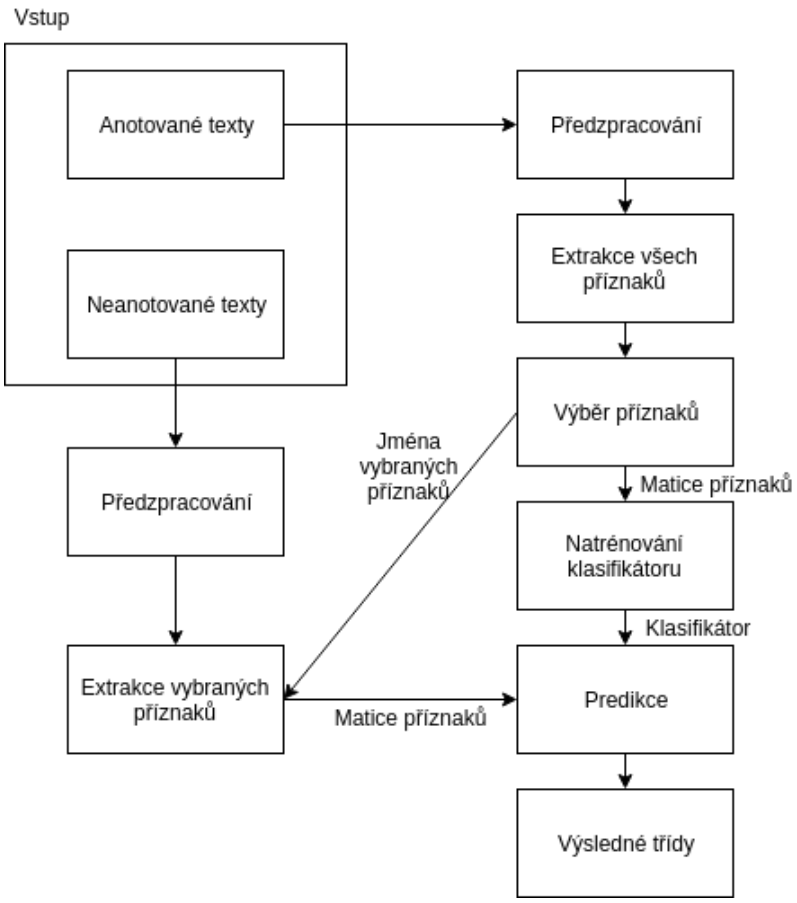
V následující sekci je popsán způsob, jakým byl program implementován, jaké byly použity metody a jaká je struktura programu.

6.2.1 Struktura programu

Program je rozdělen do několika modulů. Schéma tohoto rozdělení je možné vidět na obrázku 6.1. Zobrazené schéma odpovídá funkci systému při predikci. Na vstupu jsou jak anotované texty, tak i neanotované texty. Nejprve se pracuje s anotovanými texty (trénovací data) a provede se předzpracování a extrakce příznaků. Poté jsou z nich vybrány nejinformativnější příznaky. Provede se znovu extrakce příznaků, ale nyní se extrahují pouze vybrané příznaky z předchozí fáze a ty se vloží na vstup klasifikátoru, který je potom natrénován. Mezitím se uloží vybrané příznaky a ty se následně extrahují i z neanotovaných textů. Tím se získá vektorová reprezentace textů, které je potřeba predikovat. Klasifikátor pak provede predikci a vrátí výsledný vektor predikovaných tříd.

Při použití druhé funkce programu (odhad přesnosti) je postup poněkud odlišný. Pracuje se pouze s anotovanými daty. Ta se rozdělí na trénovací a testovací sadu. Trénovací sada se poté použije stejně jako anotovaná data při predikci a pomocí testovacích dat se spočítá přesnost modelu. Nakonec se výsledky zobrazí na výstupu.

[3] http://www.numpy.org/



Obrázek 6.1: Schéma práce programu

6.2.2 Moduly programu

Na schématu 6.1 je vidět, že program se skládá z řady modulů, které na sebe navazují. V následující sekci jsou postupně popsány.

Předzpracování

V tomto modulu jsou implementovány metody pro předzpracování textů. Vstupem i výstupem těchto metod je text. Cílem je odstranit redundantní či rušivé informace z dat. Metody jsou následující:

- Odstranění stopslov: v daném jazyku má program k dispozici seznam stopslov, která jsou následně odstraňovány z textů.
- Nahrazení čísel tokenem: jsou vyhledána všechna čísla a nahrazeny řetězcem `[num]`, který reprezentuje informaci, že na tomto místě bylo nějaké číslo.
- Nahrazení odkazů tokenem: podobný princip jako u čísel, ale nahrazují se odkazy, a to řetězcem `[url]`.
- Převod na malá písmena: všechna velká písmena jsou zaměněna za malá.
- Stematizace: slova jsou redukována na jejich kmen. Pro daný jazyk je nutné mít stematizér.

Extrakce příznaků

Tento modul provádí extrakci a výběr příznaků. Zde se provádí převod z textové reprezentace na vektorovou reprezentaci. Extrahují se nejčastěji se vyskytující slova, ale předtím se provede filtrace. Jsou odstraněny příznaky, které se vyskytují ve dvou a méně vzorcích. Stejně tak jsou odstraněny příznaky, které se vyskytují ve více než 95 % vzorků. Tím se zabrání, aby byly vybrány příznaky, které mají malou diskriminativní sílu. Modul potom vrátí matici, kde řádky představují jednotlivé vzorky a sloupce odpovídají extrahovaným příznakům.

Pro výběr příznaků se používají algoritmy popsané v kapitole 3.4. Parametrizovaná je použitá metoda výběru a počet příznaků, které se mají vybrat.

Klasifikace

Pro klasifikaci je možné použít tři modely popsané v sekci 4. Jedná se o SVM, náhodné lesy a multinomický Bayesovský klasifikátor. Pro SVM byla použita lineární jádrová funkce, která dávala lepší výsledky než RBF, a parametr C byl nastaven na hodnotu 2. Pro náhodné lesy bylo použito 128 stromů a výpočet informačního přínosu na základě gini indexu.

Výsledky

V tomto modulu se provádí výpočet výstupů z klasifikovaných dat. Funkce zde mají na vstupu obvykle predikované třídy a reálné (správné) třídy. Na základě těchto údajů potom počítají:

- celková přesnost modelu,

- přesnost po odstranění části vzorků, jejichž zařazením je si klasifikátor méně jistý,
- frekvence tříd,
- přesnost v rámci jednotlivých tříd.

Analyzátor

Zde se provádí analýza dat spočítaných v modulu výsledky. Umožňuje uložit data z více běhů programu (například z k-násobné křížové validace). Uložené výsledky se zprůměrují a vypíší se na výstup.

6.2.3 Funkce programu

Program má dvě hlavní funkce. První z nich je predikce, kde výstupem je dvojice text a predikovaná hodnota. Druhým je výpis statistik, kde program provede testy na trénovací sadě s cílem zjistit přesnost predikce a vhodnost aktuálního rozložení tříd. Na základě výsledků je možné upravit nastavení programu nebo upravit rozdělení datové sady do tříd.

Na obrázku 6.2 je možné vidět, jak statistiky vypadají. Na prvním řádku je celková přesnost. Přesnost se počítá pomocí k-násobné křížové validace. Trénovací sada se rozdělí na k částí a následně proběhne k iterací, kde v každé vždy jedna část slouží jako testovací sada a zbytek jako trénovací sada.

Následuje tabulka nazvaná *recall accuracy*, která popisuje přesnost v závislosti na daném pokrytí (accuracy to recall). Údaj 0 % znamená, že nejsou žádné vzorky odstraněny a přesnost je stejná jako u celkové přesnosti. Jak se vzorky začínají odstraňovat, tak se přesnost zvyšuje. První sloupec značí podíl odstraněných vzorků a druhý sloupec přesnost v procentech. Tato tabulka umožňuje určit procento vzorků dat, které je vhodné odstranit, aby se získal co nejlepší podíl klasifikovaných dat při vysoké přesnosti. Další tabulka *class accuracy* poskytuje zpětnou vazbu k rozdělení datové sady do tříd. V prvním sloupci jsou třídy datové sady. Druhý sloupec udává přesnost v rámci odpovídající třídy vzorků. Pokud je dosaženo hodnoty 100 %, tak všechny vzorky, které náleží dané třídě, byly klasifikovány správně. Ve třetím sloupci je možné zjistit počet trénovacích vzorků dané třídy. To je užitečné vědět, protože přesnost třídy s vysokým počtem vzorků je důležitější, než přesnost třídy s několika málo vzorky. V posledním sloupci nazvaném *frequent mislabels* jsou uvedeny dvě nejčastější třídy, u kterých dochází při klasifikaci k záměnám se třídou v prvním sloupci. Za pomlčkou je uveden počet těchto mylných predikcí, které je možné přímo porovnat s celkovou velikostí třídy ve třetím sloupci.

Program nabízí i několik sekundárních funkcí, které mají za cíl zefektivnit užívání programu. Prvním je odstranění části nejméně jistých vzorků z výsledků predikce. To umožňuje zvýšit přesnost predikce za cenu menšího počtu klasifikovaných vzorků. Druhým je odstranění jedné třídy z výsledků. Díky tomu je možné například odstranit „sběrnou“ třídu, do které se dávají vzorky, u kterých není jasné jejich zařazení. Další funkcí je aplikace převzorkování pro vyvážení datové sady.

Pro určité metody předzpracování program potřebuje znát jazyk textů, se kterými pracuje. Tuto informaci je možné zadat parametrem, ale pokud to tak uživatel neudělá, tak se program pokusí provést detekci samostatně. Využívá se k tomu knihovna googletrans[4], která umí na základě služby Google Translate detekovat jazyk textu. Jelikož datová sada

[4] https://pypi.org/project/googletrans/

```
Accuracy: 82.81%

Recall accuracy:
% of samples removed            accuracy
0%                              83%
10%                             87%
20%                             90%
30%                             93%
40%                             95%
50%                             97%
60%                             98%
70%                             99%
80%                             100%
90%                             100%

Class acccuracy:
name                    accuracy        size    frequent mislabels
-------------------------------------------------------------------------
?:                      19%             14.2    advert-8.2, gambling-1.4
koin:                   52%             3.4     bank-0.8, advert-0.4
trxid:                  60%             1.6     bank-0.2,
comp-gen:               65%             7.2     advert-0.8, bank-0.6
pelanggan:              70%             1.4     contact-0.2,
gambling:               76%             16.6    advert-3.8, bank-0.2
id:                     76%             8.4     bank-0.4, advert-0.2
advert:                 81%             47.6    gambling-4.6, ?-1.4
won:                    83%             8.0     advert-1.4,
WIB:                    89%             9.4     bank-0.6, advert-0.4
a/n:                    92%             10.2    advert-0.6, WIB-0.2
bank:                   93%             44.6    advert-1.8, WIB-0.2
tramigo:                98%             10.8    advert-0.2,
pmt:                    100%            5.6
sex:                    100%            8.4
contact:                100%            2.8
```

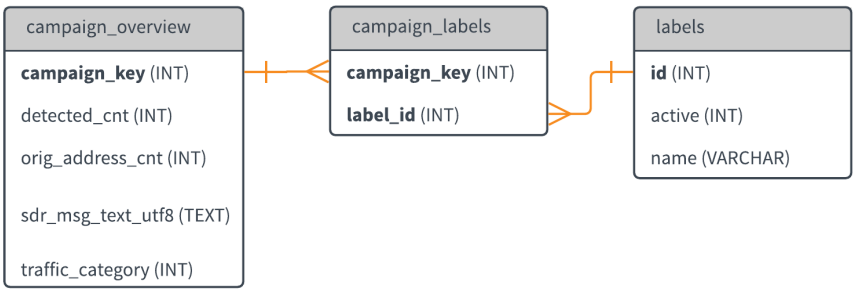
Obrázek 6.2: Příklad výstupních statistik

může obsahovat i texty ve více jazycích, tak se z ní vybere několik vzorků a provede se detekce. Jazyk datové sady se potom vybere většinovým hlasováním.

6.2.4 Vstup a výstup

Program nabízí několik možností, jak načíst vstupní data a jak uložit výsledky. První z nich je výstup do souboru ve formátu csv podle zadání. Při načítání se předpokládá, že tento soubor obsahuje alespoň dva sloupce. Jeden je pojmenován *text* a jeho položky obsahují texty SMS kampaní. U druhého sloupce je možné jeho název nastavit parametrem (pokud není zadán, tak se předpokládá jméno *label*) a ten obsahuje příslušné třídy.

Druhým možným vstupem a výstupem je databáze. Firma Mavenir má uložená data kampaní v MySQL databázi. Jako menší rozšíření oproti zadání jsem implementoval propojení mého programu s produkční databází. Důležité pro účely klasifikace byly zejména tři tabulky, jejichž vztah je znázorněn na ERD diagramu 6.3. V tabulce *campaign_overview* jsou uložena textová data. Podstatná je hlavně položka *sdr_msg_text_utf8*, kde je uložen text zprávy. Třídy jsou uloženy v tabulce labels, kde v položce *name* je uložen název dané třídy. Tyto dvě tabulky jsou spojeny asociativní tabulkou *campaign_labels*, do které se ukládá, jaké třídy patří k jakým kampaním. Jedna kampaň může mít přiřazeno více tříd. Třídy ale nemají žádné definované kategorie nebo hierarchii. Proto při načtení tříd z da-



Obrázek 6.3: ERD schéma relevantní části databáze

tabáze všechny třídy patřící k jedné kampani zkombinuji do jedné třídy, se kterou potom program pracuje.

Alternativou je načíst klasifikátor uložený na disku. Pokud se rovnou načte klasifikátor, tak není potřeba načítat trénovací data. Uložit je potřeba nejen samotný klasifikátor, ale i vybrané příznaky na základě kterých byl klasifikátor natrénován. Pokud by k tomu nedošlo, tak by program mohl vybrat jiné příznaky a přesnost klasifikátoru by mohla být odlišná od původního modelu, který byl na disk uložen. Ukládání a načítání je implementováno pomocí knihovny joblib[5], která umožňuje serializaci objektů jazyka Python na disk. Klasifikátor a příznaky se uloží do dvou souborů a mohou z nich být posléze načteny.

[5] https://github.com/joblib/joblib

Kapitola 7

Výsledky experimentů

V následující kapitole jsou popsány výsledky experimentů prováděných s vytvořeným systémem s cílem zjistit nejvhodnější metody pro klasifikaci kampaní.

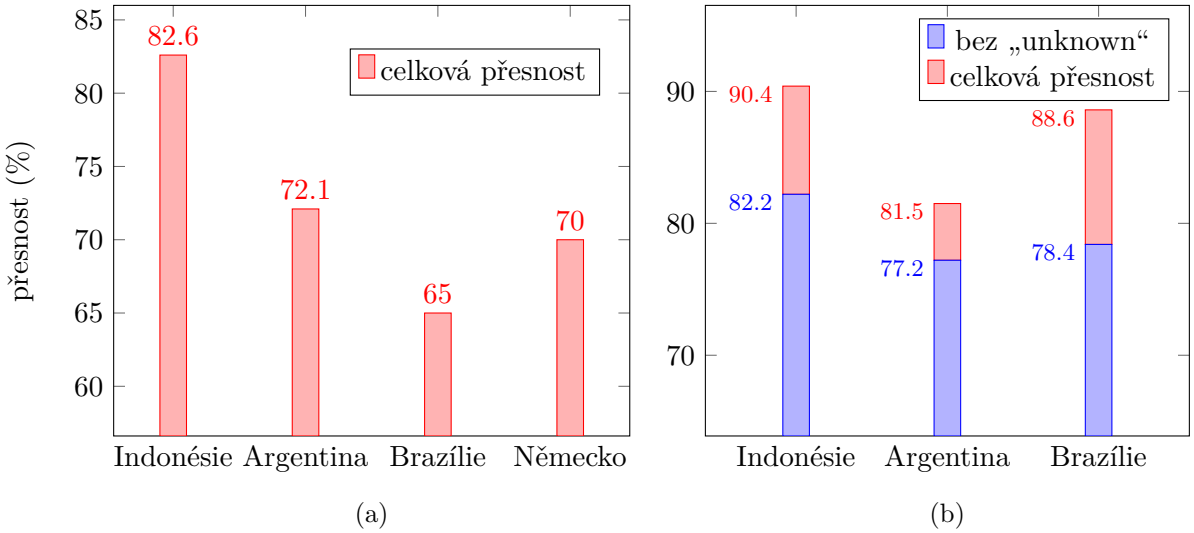
Testování proběhlo na čtyřech datových sadách, každá s tisícovkou vzorků (textů). Přesnost byla vypočítána pomocí pětinásobné křížové validace, kde se datová sada rozdělila na pět částí, čtyři z nich se použily pro natrénování klasifikátoru a zbylá pro odhad přesnosti. Tento proces se opakuje pětkrát, kde v každé iteraci se jako testovací použije jiná část. Měří se přesnost klasifikace, což je podíl správně klasifikovaných vzorků vzhledem k celkovému počtu vzorků.

Pokud nebude v textu řečeno jinak, bude použito následující nastavení:

- klasifikátor: náhodný les,
- typ extrakce: slovní n-gramy,
- maximální počet příznaků pro extrakci: 1000,
- metoda výběru příznaků: gini index,
- maximální počet vybraných příznaků: 250,
- předzpracování:
  - odstranění stopslov zapnuto,
  - nahrazení čísel vypnuto,
  - nahrazení odkazů vypnuto,
  - převod na malá písmena zapnuto,
  - stematizace vypnuta,
- vyvážení tříd vypnuto.

7.1 Přesnost klasifikace

Na grafu 7.1 je vidět přesnost klasifikace na všech datových sadách. Celkem mám k dispozici čtyři: argentinskou, brazilskou, německou a indonéskou. U německé sady není k dispozici kategorie podle subjektů, protože jsem usoudil, že se tam vyskytuje příliš málo zpráv,



Obrázek 7.1: Přehled celkové přesnosti datových sad. Na grafu 7.1a je přesnost pro kategorie podle objektu a na 7.1b podle subjektu. Na grafu 7.1b modrá část grafu značí přesnost bez třídy „unknown“ a červená část značí celkovou přesnost se všemi třídami.

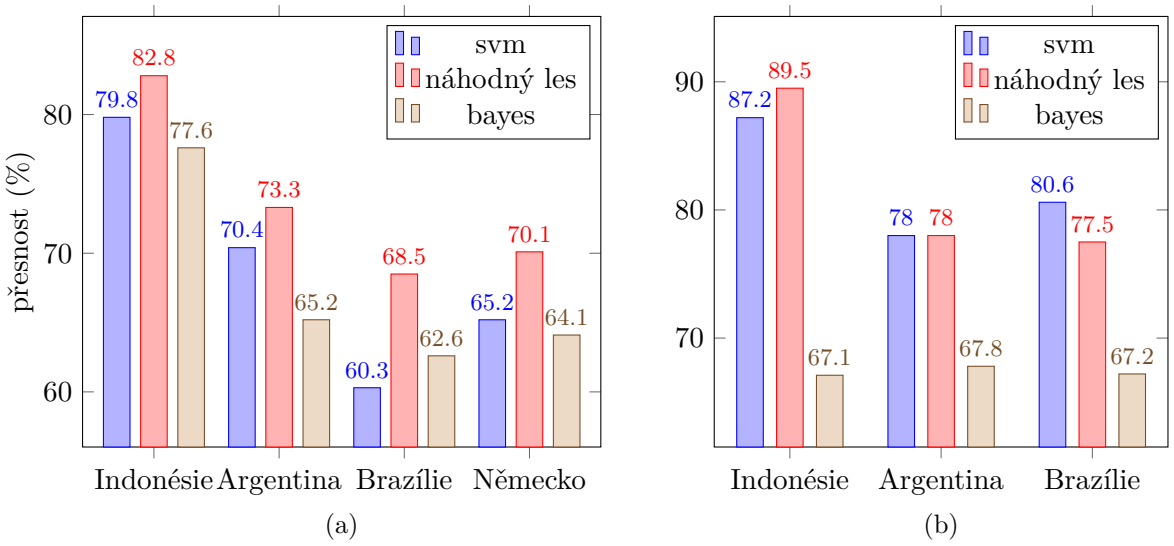
které by bylo možné přiřadit nějakému subjektu. Výrazná většina zpráv by tedy skončila v kategorii neznámých subjektů a klasifikace by nebyla moc užitečná.

Přesnost se pohybuje pro kategorie podle oboru mezi 65 %–82 % a pro kategorie podle subjektů mezi 81 %–90 %. Důvodem vyšší přesnosti kategorií podle subjektů je hlavně vysoká přesnost určení majoritní třídy – *unknown*. U většiny zpráv nelze z textu určit, kdo danou kampaň zasílá a těmto zprávám je přiřazena výše zmíněná kategorie. Její vliv je možné na pravém grafu 7.1a vidět jako rozdíl celkové přesnosti a přesnosti bez „unknown“. Dalším důvodem dobré přesnosti při kategorizaci podle subjektů je užší vazba mezi textem a kategorií. Často se dá kategorie přímo odvodit od určitého slova z textu. Naopak opačný vliv má vyšší počet tříd u kategorií podle subjektu. Pokud například zpráva obsahuje slovo *google* a jedna z kategorií sbírá zprávy od firmy Google, tak je pro klasifikátor jednoduché určit správnou třídu. Je vysoká pravděpodobnost, že zpráva, která obsahuje přímo název firmy, patří do příslušné kategorie patřící dané firmě. Naopak pokud zpráva žádný název firmy neobsahuje, určení subjektu obtížnější a je pravděpodobné, že zpráva skončí v třídě neznámých subjektů.

Nejlepších výsledků bylo dosaženo u indonéské datové sady. Tam je několik jasně definovaných kategoriích, mezi kterými se lépe rozlišuje. Naopak nejtěžší pro návrh kategorií byla německá datová sada, kde bylo velké množství zpráv obtížně zařaditelných do nějaké konkrétní skupiny. Zprávy často měly hlavně informativní charakter a týkaly se velkého množství různých témat. Německá sada také obsahovala nejméně zpráv s kampaněmi a větší podíl zaujímaly zprávy osobního charakteru.

7.2 Klasifikátory

Na obrázku 7.2 je zobrazena přesnost implementovaných klasifikátorů na argentinské datové sadě. SVM a náhodné lesy mají podobné výsledky, ale náhodné lesy mají poněkud vyšší přesnost (o 1–4 %). Naopak Bayesovský klasifikátor zaostává za ostatními výrazněji. Tyto



Obrázek 7.2: Přehled přesnosti klasifikátorů. Na grafu 7.2a je přesnost pro kategorie podle oboru a na grafu 7.2b podle subjektu.

výsledky jsou v souladu s výsledky na ostatních datových sadách. Náhodné lesy podávají stabilně nejlepší výsledky, SVM bývá na druhém místě s větším či menším odstupem.

7.3 Metody extrakce

Program podporuje tři metody extrakce příznaků – písmenné n-gramy, slovní n-gramy a slovní skipgramy. Na grafech v obrázku 7.3 je zobrazena přesnost těchto metod. Graf 7.3a zobrazuje data pro kategorie podle oboru a graf 7.3b pro kategorie podle subjektu. Pro slovní n-gramy jsou použity rozsah n mezi 1 až 3. U písmenných n-gramů jsem použil rozpětí mezi dvěma a osmi znaky. U skipgramů jsem zvolil bigramy.

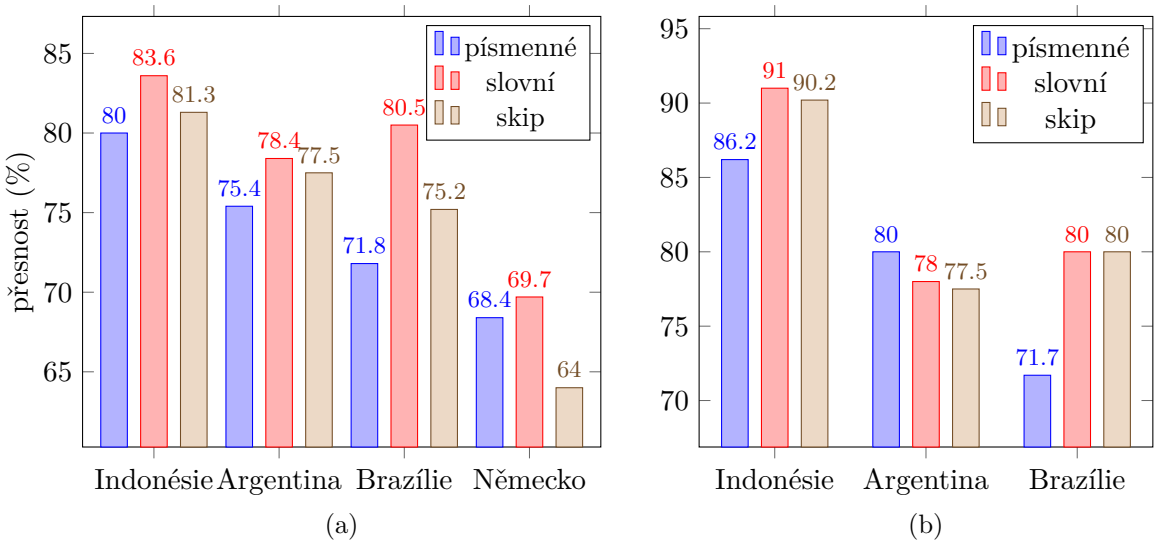
Slovní n-gramy mají vyšší rozsah textu, který mohou zaznamenat v jednom příznaku (konkrétnější příznaky), ale na takto krátkých textech jsou vyšší hodnoty n obvykle nepoužitelné a tak naprostá většina příznaků jsou unigramy. To je vidět i u skipgramů, které musí být minimálně bigramy. Jejich přesnost je konzistentně nižší než u slovních n-gramů.

Na grafech je vidět, že slovní n-gramy podávají stabilně nejlepší výsledky. Skipgramy jsou obvykle na druhém místě, ale v některých případech jsou překonány písmennými n-gramy.

7.4 Metody předzpracování

V programu je implementováno pět metod předzpracování – odstranění stopslov, nahrazení čísel, nahrazení odkazů, převod na malá písmena a stematizace. Nejdříve jsou zkoumány metody předzpracování jednotlivě a potom jsou vybrány ty s pozitivním vlivem na přesnost a vyzkouší se jejich kombinace.

Na grafu 7.4a je zobrazen přínos jednotlivých metod předzpracování oproti základní přesnosti, kde jsou všechny kroky předzpracování vypnuty, pro argentinské datové sadě. K metodám s pozitivním vlivem na přesnost patří odstranění stopslov, převod na malá



Obrázek 7.3: Přesnost slovních a písmenných n-gramů a skipgramů. Graf 7.3a popisuje přesnost při použití kategorií podle oboru a graf 7.3b podle subjektu.

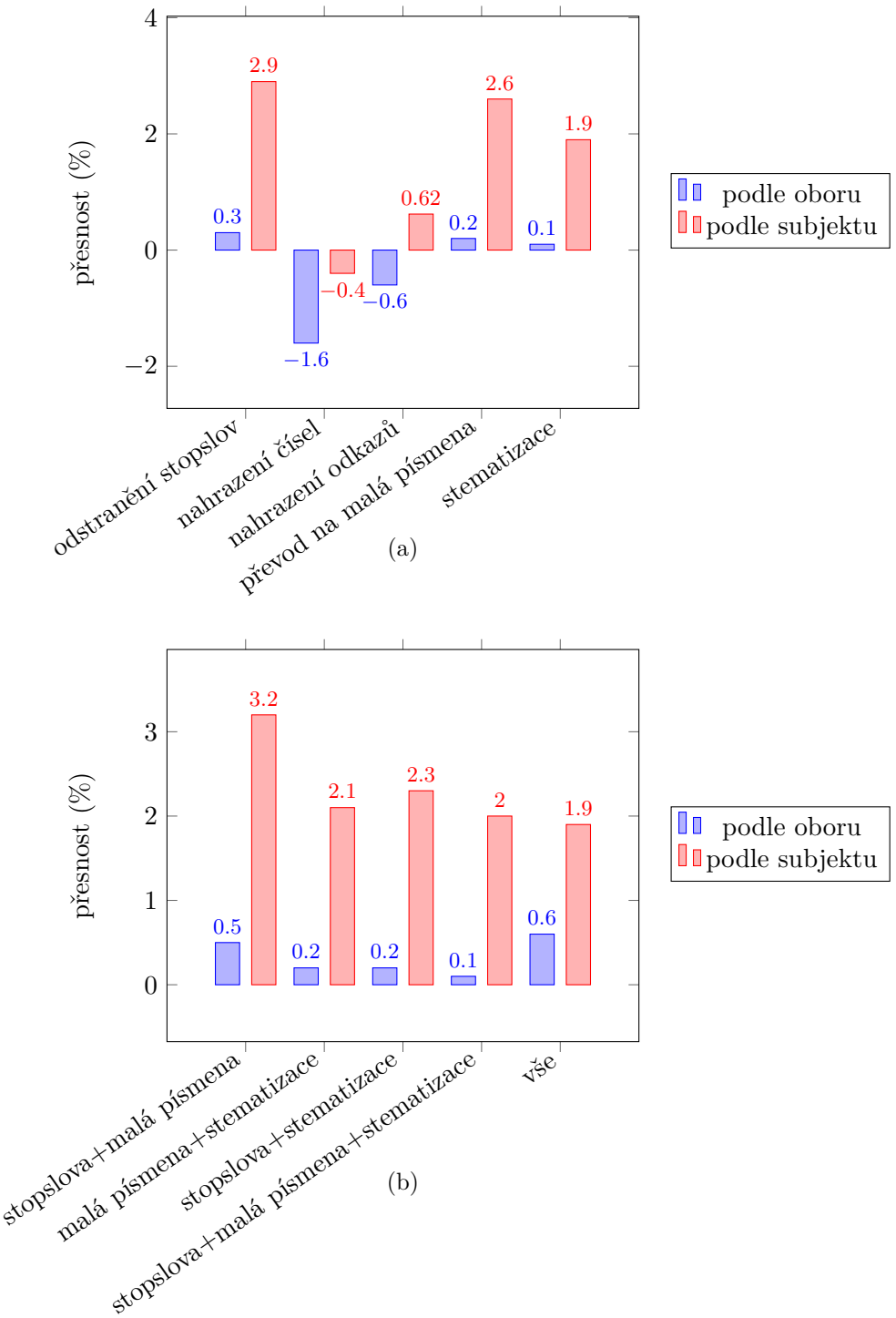
písmena a stematizace. Naopak k těm, které mají na přesnost neutrální nebo negativní vliv, patří nahrazení čísel a odkazů.

Nahrazení odkazů nemá jednoznačně pozitivní vliv. Pro kategorie podle subjektu se přesnost mírně zvýší a pro kategorie podle oboru lehce klesá.

Ani odstranění čísel nemá pozitivní efekt. Číselné údaje v textech zpráv se vyskytují zejména jako telefonní čísla a dále jako částky výher a dluhů. Samotné částky jsou ale značně variabilní. Telefonní čísla se opakují a je možné na základě nich určit třídu zprávy. Často několik kampaní s různým textem vyzývá příjemce, aby zavolal na stejné číslo. V porovnání s odkazy jsou ale tyto skupiny menší. Různé druhy čísel mohou pomoci odlišit typy zpráv podle oboru. Například pokud je ve zprávě telefonní číslo nebo nějaká finanční částka, tak se velmi často jedná o výzvu k zaplacení dluhu.

Pozitivní vliv převodu na malá písmena bylo možné očekávat, protože tento krok eliminuje variabilitu, kterou do textu zanesou například začátky vět, ale nemá to vliv na samotný obsah textu, který zůstane zachován. Stejně odstranění stopslov odstraňuje pouze slova neobsahová (spojky, předložky...) a nezmění se obsahová slova (podstatná a přídavná jména, slovesa), na základě kterých se potom určuje kategorie textu. Odstranění stopslov má relativně vysokou a vyrovnanou úspěšnost pro oba typy kategorií. Stematizace zkrátí slova na základní kmen a tím umožňuje odstranit variabilitu vzniklou skloňováním a časováním. Pro klasifikační algoritmus není důležité, v jakém přesném tvaru se dané slovo v textu nachází, ale je podstatné, zda se ve zprávě dané slovo nachází nebo nenachází. Díky stematizaci je možné tuto informaci uchovat v menším počtu příznaků.

Na grafu 7.4b je zobrazen vliv kombinace více metod předzpracování. Zahrnul jsem pouze ty kombinace, které samostatně vykazovaly zlepšení oproti základní přesnosti. Na první pohled je vidět, že kategorie podle subjektu profitují z aplikace předzpracování mnohem více, než kategorie podle oboru. To platí jak pro jednotlivé metody, tak pro jejich kombinace. Z grafu je patrné, že nejlepší výsledky produkuje kombinace stopslova + malá písmena, i když rozdíly nejsou velké. K té jsem se taky přiklonil pro použití v programu. Výhodou této kombinace je také fakt, že je méně závislá na použitém jazyku. Sice je nutný



Obrázek 7.4: Vliv metod předzpracování na přesnost klasifikace na argentinské datové sadě. Graf 7.4a popisuje změnu přesnosti při použití kategorií metod samostatně a graf 7.4b ukazuje změnu přesnosti při použití kombinace více metod.

seznam stopslov pro použitý jazyk, ale ten je mnohem méně obtížné získat než stematizér pro daný jazyk. Metoda převodu na malá písmena je potom jazykově nezávislá, pokud jazyk používá latinku. Pro odlišné druhy písem by bylo nutné tuto metodu upravit.

I když vztahy mezi jednotlivými kategoriemi zůstávají zachované i u ostatních datových sad, tak absolutní míra zvýšení přesnosti se může výrazně lišit. Brazilská datová sada má konzistentně nejvyšší zvýšení přesnosti, kde se změna pro kategorie podle oboru pohybuje v rozmezí 2–4 % a pro kategorie podle subjektu kolem 5 %. Naopak u indonéské a německé datové sady lze pozorovat nižší zvýšení přesnosti v rozmezí 2 % pro oba typy kategorií.

Pro kategorie podle subjektu mají metody předzpracování obecně pozitivní vliv, zatímco pro kategorie podle oboru je přínos výrazně nižší. Tři metody předzpracování (stopslova, stematizace, malá písmena) mají rozhodně pozitivní vliv na přesnost klasifikace a jejich aplikace je velice rychlá. Jediným problémem je závislost části metod na použitém jazyku. Odstranění stopslov vyžaduje seznam stopslov, stematizace vyžaduje příslušný stematizér. Pokud jsou ale tyto prostředky k dispozici, tak je výhodné zmíněné tři metody využít.

7.5 Metody výběru příznaků

Program podporuje čtyři metody – gini index, informační přínos, chí kvadrát a výběr na základě společné informace. Na obrázku 7.5 je zobrazen vliv metod výběru příznaků na přesnost klasifikace pro argentinskou datovou sadu. Ostatní datové sady vykazují podobné závislosti. Testování bylo prováděno na klasifikátoru SVM, protože metody gini index a informační přínos jsou založeny na náhodných lesích a použití stejného modelu i při klasifikaci by mohlo výsledky vychýlit ve prospěch těchto metod.

Nejlepší výsledky poskytuje metoda gini index, která zejména pro kategorie podle subjektů má výrazně vyšší přesnost než ostatní metody. Pro kategorie podle oboru jsou gini index a informační přínos srovnatelné. Metoda chí kvadrát a metoda společné informace mají při aplikaci na kategorie podle oboru dokonce negativní vliv na přesnost. Obecně se dá z výsledků vyvodit závěr, že metody založené na náhodných lesích dávají lepší výsledky než metody statistické.

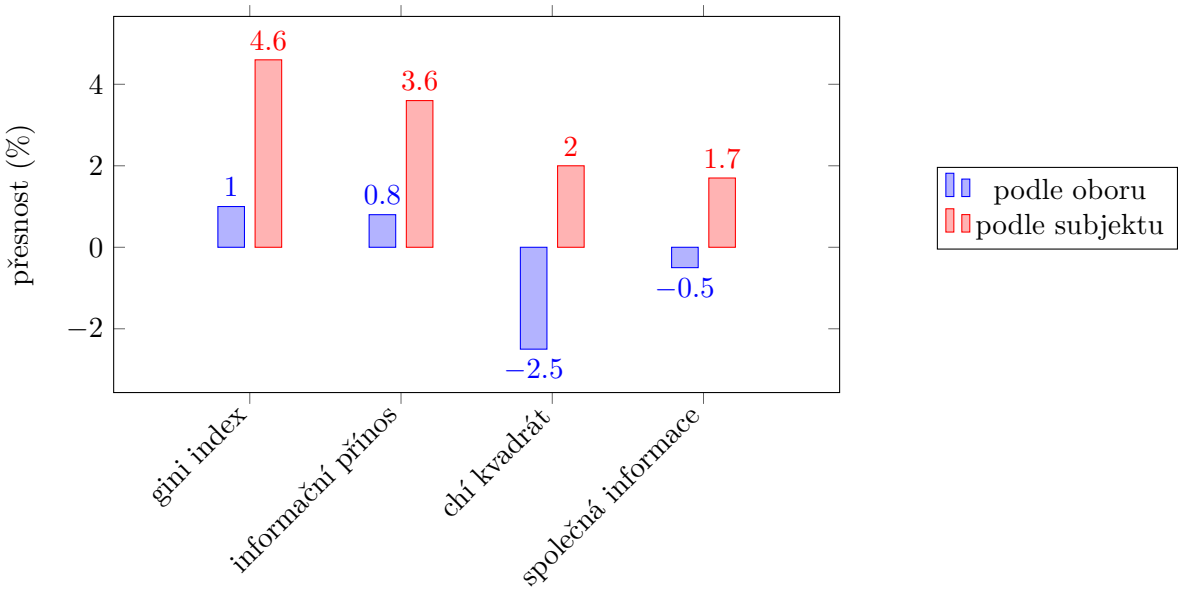
Zajímavý je rozdíl zlepšení při aplikaci výběru příznaků mezi kategoriemi tříd. Pro kategorie podle subjektu je zlepšení výrazné u všech metod. Naopak u kategorií podle oboru je spíše mírné, případně může být změna i negativní. To odpovídá výsledkům pro metody předzpracování, kde se pro kategorie podle subjektu také dosahuje lepších výsledků.

Byly vyzkoušeny i transformační metody jako je PCA (Principal Compenent Analysis), LDA (Latent Dirichlet Allocation) a NMF (Non-Negative Matrix Factorization), ale ty nedávaly lepší výsledky než metody popsané výše.

Metody výběru příznaků jsou užitečné a mají obecně pozitivní vliv na přesnost klasifikace, i když zvýšení přesnosti není výrazné. Zmíněné metody mají nezanedbatelný podíl na době běhu programu a proto při zpracování většího množství dat může být vhodné metody výběru příznaků vypnout za účelem urychlení programu. Další diskuze na toto téma je v kapitole 7.9, která se zabývá dobou běhu programu. Je třeba zvážit, že zvýšení přesnosti na jiné datové sadě může být vyšší.

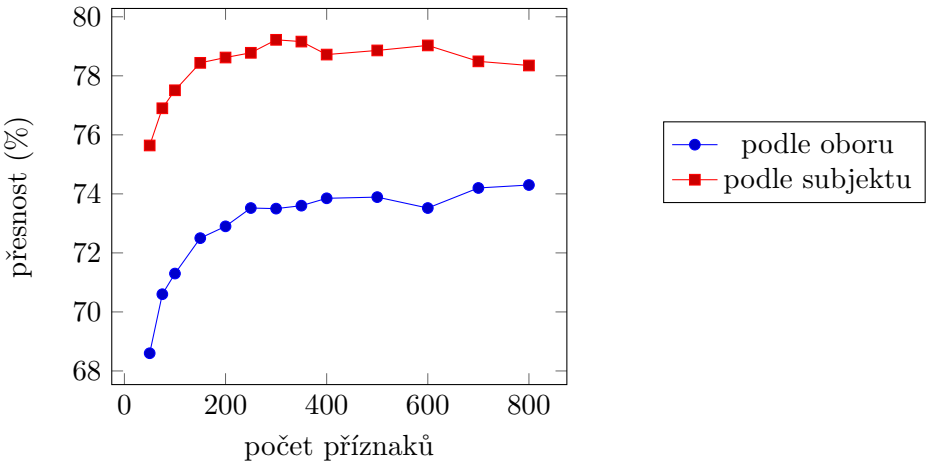
7.6 Počet příznaků

Počet příznaků má významný vliv na přesnost klasifikace i dobu běhu programu. Je proto potřeba určit vhodný kompromis mezi snahou zvýšit přesnost (vyšší počet příznaků) a



Obrázek 7.5: Vliv metod výběru příznaků oproti základní přesnosti, kdy je výběr příznaků vypnut.

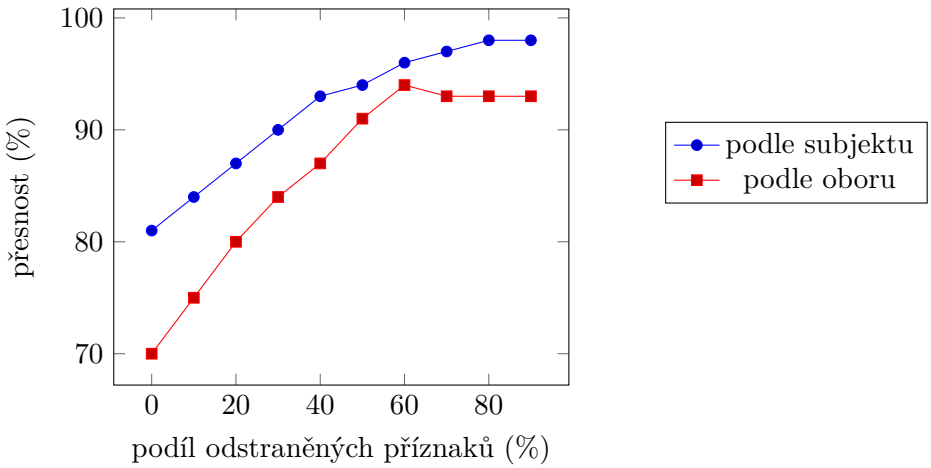
snahou zrychlit dobu výpočtu (nižší počet příznaků). Na grafu 7.6 je zobrazena závislost počtu příznaků na přesnosti pro argentinskou datovou sadu. Přesnost prudce stoupá až do počtu příznaků 150–200. Poté ještě mírně stoupá do počtu 250 příznaků. Následně se přesnost už nijak zásadně nezvyšuje. Z toho důvodu jsem zvolil limit počtu příznaků 250.



Obrázek 7.6: Vliv počtu příznaků na přesnost klasifikace

7.7 Pokrytí

Pokud se umožní systému neklasifikovat vzorky, jejichž zařazením si je méně jistý, tak je možné zvýšit přesnost za cenu, že ne všechny vzorky budou klasifikované. Podíl vzorků, které byly nakonec klasifikovány vzhledem k celkovému počtu, budu nazývat pokrytí.



Obrázek 7.7: Závislost přesnosti na podílů odstraněných vzorků

Klasifikační algoritmus může ke každému vzorku přiřadit ke každé možné třídě procentuální míru pravděpodobnosti, která značí, do jaké míry si klasifikátor myslí, že vzorek patří do určité třídy. Metoda počítá podíl jistot dvou tříd s nejvyšší pravděpodobností podle vzorce 7.1, kde k je vypočítaný koeficient, který se porovnává s nastavenou hranicí, a je skóre nejpravděpodobnější třídy a b je skóre třídy s druhou nejvyšší pravděpodobností. Cílem je zjistit, jaký je odstup nejpravděpodobnější varianty od druhé nejpravděpodobnější. Čím je vyšší, tím je jistější, že se jedná o správnou třídu.

$$k = 1 - \frac{b}{a} \tag{7.1}$$

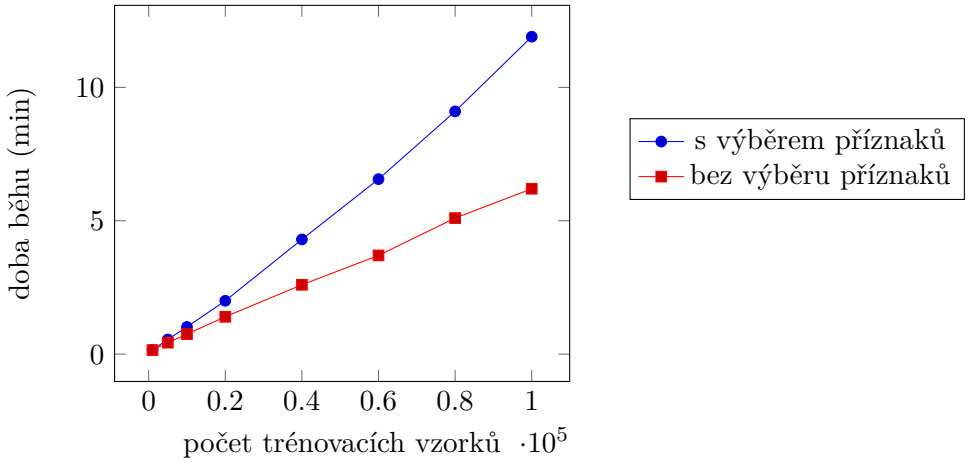
Na obrázku 7.7 je zobrazena závislost přesnosti vzhledem k procentuálnímu podílu odstraněných vzorků. Je vidět, že při malém podílu odstraněných vzorků (10–20 %), je možné znatelně zvýšit přesnost klasifikace (6–10 %). Odstraňovat větší množství vzorků se už ale nevyplatí – zvyšování přesnosti se zpomaluje a čím více vzorků je odstraněno, tím jsou výsledky méně užitečné.

7.8 Vyvážení dat

Problematickým atributem datových sad, se kterými v této práci pracuji, je značná nevyváženost tříd. To jsem zkoušel vyřešit metodami vyvážení dat popsanými v kapitole 3.3. Přínos těchto metod však nebyl dostatečný, aby to převážilo negativní aspekty. U převzorkování byl problém s velkým nárůstem velikosti datové sady, čímž se výrazně prodloužila doba trénování. Naopak u podvzorkování byl problém s velkým snížením velikosti trénovací datové sady a z toho plynoucím nedostatkem vzorků pro naučení modelu. Proto je v základním nastavení vyvážení dat vypnuto.

7.9 Škálovatelnost

Škálovatelnost systému je stejně důležitá jako jeho přesnost, protože pokud běh programu trvá příliš dlouho, tak se snižuje jeho použitelnost. Na grafu 7.8 je zobrazena závislost počtu vzorků v trénovací sadě na době běhu programu. Testování proběhlo na procesoru



Obrázek 7.8: Doba trénování v minutách v závislosti na velikosti trénovací sady (ve statisících)

Intel i5 6600. Je velice pozitivní, že závislost je lineární a tedy se s přidáváním dat nezvyšuje neúměrně doba běhu programu. Program zvládne natrénovat 100 000 vzorků za 12 minut. To je dobrý výsledek zejména protože není nutné, aby program dával výsledky na výstup v reálném čase. Stačí pokud výpočet netrvá příliš dlouho a v určitých intervalech se výsledky mohou přepočítávat. Alternativně je možné natrénovat klasifikátor jen jednou a výsledek uložit na disk. Následné načtení klasifikátoru je v porovnání s dobou potřebnou pro trénování zanedbatelné. Také doba potřebná pro predikci je oproti délce trénování nízká. Predikce 500 000 vzorků trvá méně než dvě minuty.

Zajímavé je srovnání rychlosti programu s vypnutým a zapnutým výběrem příznaků. Doba, kterou výběr příznaků zabere je znatelná a rozdíl se rychle zvětšuje se zvětšováním trénovací sady. Jelikož doba běhu systému i se zapnutým výběrem příznaků je přijatelná, tak jsem ponechal tuto funkci v základním nastavení zapnutou. Pro výrazně větší datové sady může být užitečné výběr příznaků vypnout.

Paměťové nároky systému pro 100 000 trénovacích vzorků byly přibližně 1,2 GB.

Kapitola 8

Závěr

Tato práce se zabývá klasifikací textových dat. Firma Mavenir potřebovala vytvořit systém, který by byl schopen klasifikovat spamové SMS kampaně do definovaných kategorií na základě jejich obsahu. Pro tento účel byly anotovány čtyři datové sady reálných spam kampaní s různými jazyky. Datové sady měly tisíce textů a z každé bylo 1000 vzorků anotováno. Jedním z výstupů je popis vhodných praktik pro ruční anotaci podobných souborů s cílem dosažení co nejvyšší přesnosti pro daný účel (určení oblasti / odesílatele SMS kampaně). Anotace má velmi výrazný vliv na výslednou přesnost klasifikace. Vhodným sloučením závislých tříd je možné přesnost klasifikátoru výrazně zvýšit. Při porovnání knihoven pro strojové učení byla pro implementaci vybrána knihovna scikit-learn (SKLearn) z důvodu její efektivity na malých a středně velkých datových sadách.

Byl implementován program pro kategorizaci kampaní. Jeho primární funkcí je predikce tříd textů na základě sady trénovacích vzorků, které je možné načíst buď ze souboru formátu csv nebo z databáze. Stejně tak výstup predikce je možné uložit do souboru nebo do databáze. Sekundární funkcí programu je odhad přesnosti klasifikace na trénovací sadě. Program spočítá přesnost pomocí k-násobné křížové validace a vypíše další statistiky, které dávají zpětnou vazbu k volbě tříd v trénovací sadě. Program podporuje metody popsané v teoretické části diplomové práce a pomocí parametrů programu je mezi nimi možné vybírat.

Dalším výsledkem je zjištění výkonosti jednotlivých metod pro předzpracování a klasifikaci a zjištění, jaké jsou vhodné pro danou aplikační doménu (krátké textové zprávy). Ukázalo se, že z metod předzpracování má pozitivní vliv na přesnost odstranění stopslov, převod na malá písmena a stematizace. Na druhé straně je u těchto metod nevýhodou určitá jazyková závislost a nejdou tedy aplikovat na text jakéhokoliv jazyka. Naopak neutrální nebo negativní vliv na přesnost mělo odstranění čísel a odkazů. Čísla i odkazy umožňují klasifikátoru přesněji určit správnou třídu. Mezi klasifikátory se ukázaly nejlepší náhodné lesy, které překonaly jak SVM, tak Bayesovský klasifikátor. Pro extrakci příznaků jsem zvolil slovní n-gramy, které mají stabilně vyšší přesnost než písmenné n-gramy a slovní skipgramy. Výběr příznaků se potvrdil jako krok s pozitivním vlivem na přesnost klasifikace. Nejlepší výsledky měla metoda Gini index, která překonala metody informační přínos, chí kvadrát a metodu společné informace.

Trénování klasifikátoru na 100 000 textů se zapnutým výběrem příznaků trvá 12 minut a zabere 1,2 GB paměti. Trénování na jedné datové sadě stačí provést jen jednou a následně je možné uložit natrénovaný klasifikátor na disk.

Celkem bylo dosaženo nejvyšší přesnosti 82 % pro rozdělení podle oborů a 90 % pro rozdělení podle subjektů. Tato přesnost se dá dále zvýšit, pokud se umožní systému nekla-

sifikovat některé vzorky. Při 80 % pokrytí se získá přesnost 91 % pro rozdělení podle oborů 94 % pro rozdělení podle subjektů.

Program je možné dále rozvíjet zejména implementací dalších metod klasifikace, předzpracování, extrakce a výběrů příznaků.

Literatura

[1] Aggarwal, C. C.; Zhai, C. X.: *Mining Text Data*. Springer Publishing Company, Incorporated, 2012, ISBN 1461432227, 9781461432227.

[2] Awad, M.; Khanna, R.: *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*. Berkely, CA, USA: Apress, první vydání, 2015, ISBN 1430259892, 9781430259893.

[3] Breiman, L.: *Classification and regression trees*. Routledge, 2017.

[4] Cortes, C.; Vapnik, V.: Support-Vector Networks. *Machine Learning*, ročník 20, č. 3, Sep 1995: s. 273–297, ISSN 1573-0565, doi:10.1023/A:1022627411411.
URL https://doi.org/10.1023/A:1022627411411

[5] Darwish, K.; Magdy, W.; Mourad, A.: Language processing for arabic microblog retrieval. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, ACM, 2012, s. 2427–2430.

[6] Denny, M. J.; Spirling, A.: Text preprocessing for unsupervised learning: why it matters, when it misleads, and what to do about it. *Unpublished manuscript, Dep. Polit. Sci, Stanford Univ and Inst. Quant. Soc. Sci., Harvard Univ. https://ssrn. com/abstract*, ročník 2849145, 2017.

[7] Elrahman, S. M. A.; Abraham, A.: A review of class imbalance problem. *Journal of Network and Innovative Computing*, ročník 1, č. 2013, 2013: s. 332–340.

[8] Ferilli, S.; Esposito, F.; Grieco, D.: Automatic Learning of Linguistic Resources for Stopword Removal and Stemming from Text. *Procedia Computer Science*, ročník 38, č. Supplement C, 2014: s. 116 – 123, ISSN 1877-0509, doi:https://doi.org/10.1016/j.procs.2014.10.019, 10th Italian Research Conference on Digital Libraries, IRCDL 2014.
URL http://www.sciencedirect.com/science/article/pii/S1877050914013799

[9] Galar, M.; Fernandez, A.; Barrenechea, E.; aj.: A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, ročník 42, č. 4, 2012: s. 463–484.

[10] Gudkova, D.; Vergelis, M.; Demidova, N.; aj.: Spam and phishing in Q1 2016. *Kaspersky Lab*, 2016: s. 1–22.

[11] Jivani, A. G.; aj.: A comparative study of stemming algorithms. *Int. J. Comp. Tech. Appl*, ročník 2, č. 6, 2011: s. 1930–1938.

[12] Kang, P.; Cho, S.: EUS SVMs: Ensemble of under-sampled SVMs for data imbalance problems. In *Neural Information Processing*, Springer, 2006, s. 837–846.

[13] Kontos, J.; Lekakis, J.; Malagardi, I.; aj.: Grammars for question answering systems based on intelligent text mining in biomedicine. In *Proceedings of the 7th Hellenic Europeoan Conference on Computer Mathematics and its Applications*, ročník 95, 2005.

[14] Ling, X.; Jiang, J.; He, X.; aj.: Generating gene summaries from biomedical literature: A study of semi-structured summarization. *Information Processing & Management*, ročník 43, č. 6, 2007: s. 1777 – 1791, ISSN 0306-4573, doi:https://doi.org/10.1016/j.ipm.2007.01.018, text Summarization.
URL http://www.sciencedirect.com/science/article/pii/S030645730700043X

[15] Liu, N.; Zhang, B.; Yan, J.; aj.: Text representation: From vector to tensor. In *Data Mining, Fifth IEEE International Conference on*, IEEE, 2005, s. 4–pp.

[16] Masuyama, T.; Nakagawa, H.: Applying cascaded feature selection to SVM text categorization. In *Proceedings. 13th International Workshop on Database and Expert Systems Applications*, Sept 2002, ISSN 1529-4188, s. 241–245, doi:10.1109/DEXA.2002.1045905.

[17] Okazaki, S.; Taylor, C. R.: What is SMS advertising and why do multinationals adopt it? Answers from an empirical study in European markets. *Journal of Business Research*, ročník 61, č. 1, 2008: s. 4–12.

[18] Platt, J.: Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines. Technická zpráva, April 1998.
URL https://www.microsoft.com/en-us/research/publication/sequential-minimal-optimization-a-fast-algorithm-for-training-support-vector-machines/

[19] Pomikálek, J.; Rehurek, R.: The Influence of preprocessing parameters on text categorization. *International Journal of Applied Science, Engineering and Technology*, ročník 1, 2007: s. 430–434.

[20] Raskutti, B.; Kowalczyk, A.: Extreme re-balancing for SVMs: a case study. *ACM Sigkdd Explorations Newsletter*, ročník 6, č. 1, 2004: s. 60–69.

[21] Rokach, L.; Maimon, O.: *Data mining with decision trees: theory and applications*. World Scientific, 2008.

[22] Shirani-Mehr, H.: SMS spam detection using machine learning approach. Technická zpráva, tech. rep., Stanford University, 2013.

[23] Song, F.; Liu, S.; Yang, J.: A Comparative Study on Text Representation Schemes in Text Categorization. *Pattern Anal. Appl.*, ročník 8, č. 1, Září 2005: s. 199–209, ISSN 1433-7541, doi:10.1007/s10044-005-0256-3.
URL http://dx.doi.org/10.1007/s10044-005-0256-3

[24] Song, F.; Liu, S.; Yang, J.: A comparative study on text representation schemes in text categorization. *Pattern analysis and applications*, ročník 8, č. 1-2, 2005: s. 199–209.

[25] Sun, Y.; Kamel, M. S.; Wong, A. K.; aj.: Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, ročník 40, č. 12, 2007: s. 3358–3378.

[26] Wang, D.; Irani, D.; Pu, C.: A study on evolution of email spam over fifteen years. In *Collaborative Computing: Networking, Applications and Worksharing (Collaboratecom), 2013 9th International Conference Conference on*, IEEE, 2013, s. 1–10.

[27] Wang, S.; Manning, C. D.: Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, Association for Computational Linguistics, 2012, s. 90–94.

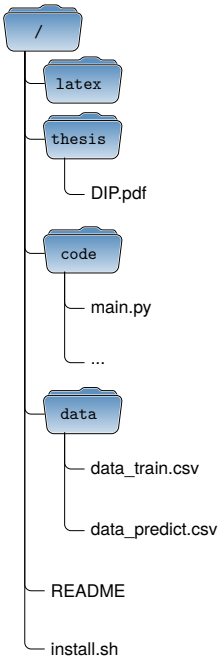
[28] Weinberger, K.; Dasgupta, A.; Langford, J.; aj.: Feature hashing for large scale multitask learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ACM, 2009, s. 1113–1120.

[29] Xu, Q.; Xiang, E. W.; Yang, Q.; aj.: Sms spam detection using noncontent features. *IEEE Intelligent Systems*, ročník 27, č. 6, 2012: s. 44–51.

Příloha A

Obsah přiloženého paměťového média

Na obrázku A.1 je zobrazena struktura odevzdávaného paměťového média. Ve složce `latex` je zdrojový kód technické zprávy v LaTeXu. Ve složce `thesis` technická zpráva ve formátu pdf. Složka `code` obsahuje zdrojové soubory vytvořené v rámci diplomové práce. Složka `data` obsahuje testovací sadu dat (SMS Spam Collection [1]). Data v testovací sadě jsou jiná než ta, která byla používána pro vyhodnocení výsledků této práce. Data reálných SMS zpráv nebylo možné zveřejnit, protože patří firmě Mavenir. Nicméně na testovací datové sadě je možné vyzkoušet všechny funkce programu. Soubor `README` obsahuje pokyny k instalaci závislostí a typické příklady použití programu. Skript `install.sh` nainstaluje potřebné knihovny (pokud je podporováno `apt-get`).



Obrázek A.1: Struktura přiloženého paměťového média

[1] http://www.dt.fee.unicamp.br/~tiago/smsspamcollection/

Příloha B

Manuál

Rozhraní programu je pojato formou konzolové aplikace. Program vyžaduje nastavení dvou skupin parametrů. Prvním přepínačem se ovládá vstup programu:

- `-f FILE` FILE je cesta k souboru s trénovacími vzorky,
- `-d [host_name] [username] [password] [database_name]` Zadává parametry pro připojení se k databázi. Program bude pro vstup a výstup využívat databázi.

Druhý přepíná mezi módem predikce a statistik.

- `-p FILE` Nastaví program pro predikci tříd vzorků v souboru FILE. Pokud je současně zadán parameter `-d`, tak se soubor FILE ignoruje.
- `-a K` Program vypíše statistiky přesnosti na vstupním trénovacím souboru pomocí K-násobné křížové validace.

Parametr `-h` vypíše na výstup nápovědu. Následující nepovinné parametry umožňují použít alternativní vstupy a výstupy.

- `-s FILE1 FILE2` Provede uložení modelu do souboru. Do FILE1 se uloží klasifikátor. Do FILE2 se uloží používané příznaky.
- `-l FILE1 FILE2` Umožňuje načtení modelu ze souboru do programu. FILE1 je soubor s uloženým klasifikátorem. FILE2 je soubor s uloženými příznaky.
- `-c LAB` Nastavuje název sloupce (LAB) ve vstupním souboru csv, kde jsou uložené třídy.
- `-o FILE` Zapíše výstup do souboru FILE, místo aby se vypsal na standardní výstup.
- `-os` Program v módu statistik (-a) vypíše pouze celkovou přesnost.

Parametry uvedené v následujícím seznamu slouží k nastavení algoritmu strojového učení a extrakce příznaků

- `-la LANG` Nastaví používaný jazyk pro metody předzpracování. Jazyk je očekáván ve formátu dvojpísmenného kódu podle standardu ISO (například `en` pro anglický jazyk). Pokud není zadán, tak se program pokusí provést detekci jazyka automaticky.

- `-ba` Zapne vyvážení trénovací datové sady, aby klasifikátor mohl lépe predikovat minoritní vzorky. Používá se náhodné převzorkování. Ve výchozím nastavení je vypnuto.
- `-fem ARG` Určuje, jaká metoda se má použít pro extrakci příznaků. Možnosti – slovní n-gramy, písmenné n-gramy, dvouprvkové skip-gramy. Výchozí nastavení: slovní n-gramy.
- `-fel NUM` Určuje maximální počet příznaků, které se mohou extrahovat. Výchozí nastavení: 1000.
- `-nofs` Vypne výběr příznaků. Ve výchozím nastavení je výběr příznaků zapnut.
- `-fsm ARG` Vybírá metodu, která se má použít pro výběr příznaků. Možné hodnoty jsou `gini`, `mutual`, `entropy` a `chi2`, což odpovídá metodám gini index, mutual info, information gain a χ-kvadrát. Výchozí nastavení: gini index.
- `-fsl NUM` Určuje maximální počet příznaků, které se mají vybrat metodou výběru příznaků. Jako NUM se očekává kladné celé číslo. Výchozí nastavení: 250.
- `-pp P1 P2 P3 P4 P5` Nastavení předzpracování. Program podporuje pět metod předzpracování. V pořadí se jedná o odstranění stopslov, nahrazení čísel, nahrazení odkazů, převod na malá písmena a stematizace. Ve výchozím nastavení zapnuto odstranění stopslov a převod na malá písmena. Pro vypnutí určité metody stačí na odpovídající místo zapsat řetězec `off` a pro zapnutí `on`
- `-tm ARG` Zvolí použitý klasifikátor. Možné hodnoty jsou `svm`, `tree`, `bayes`. Což odpovídá klasifikátorům SVM, náhodný les a Bayesovskému klasifikátoru. Výchozí nastavení: náhodný les.
- `-ru NUM` Odstraní podíl nejméně jistých vzorků, odpovídajících číslu ARG, z výsledků predikce. Jako parametr přijímá desetinné číslo v rozmezí (0,1).
- `-rc CLS` Odstraní třídu CLS z výpočtu výsledků. CLS je jméno třídy, která se má odstranit.

B.1 Typické použití

V následující sekci popíšu typické použití programu. Následující příkaz spustí program, který natrénuje klasifikátor na datech ze souboru `./data/argentina.csv` a následně vypíše statistiky přesnosti, která je spočítána pomocí 5-násobné křížové validace.

```
./main.py -f ./data/argentina.csv -a 5
```

Druhou důležitou funkcí je predikce. Následující příkaz natrénuje klasifikátor stejně jako v předchozím případě, ale následně se spustí predikce nad texty ze souboru `./data/arg_predict.csv`.

```
./main.py -f ./data/arg.csv -p arg_predict.csv
```

Na dalším příkladě ukážu, jak je možné uložit a načíst klasifikátor. Na prvním řádku se podobně jako na předchozích příkladech natrénuje klasifikátor na souboru. Následně se pomocí příkazu `-s` uloží do dvou souborů. Z těch je potom možné klasifikátor načíst jak je ukázáno na druhém řádku. V tomto případě není potřeba zadávat parametr `-f`, protože klasifikátor už je natrénovaný a stačí ho jen načíst.

```
./main.py -f file1.csv -p file2.csv -s classif.pkl features.pkl
./main.py -p file2.csv -l classif.pkl features.pkl
```